Complexity Theoretic Aspects of Rank, Rigidity and Circuit Evaluation

By

Jayalal Sarma M.N.

THE INSTITUTE OF MATHEMATICAL SCIENCES,
CIT CAMPUS, TARAMANI, CHENNAI.

A thesis submitted to the
Board of Studies in Mathematical Sciences

In partial fulfillment of the requirements
For the Degree of

DOCTOR OF PHILOSOPHY

of

HOMI BHABHA NATIONAL INSTITUTE

February 2009

Homi Bhabha National Institute

Recommendations of the Viva Voce Board

As members of the Viva Voce Board, we recommend that the dissertation prepared by **Jayalal Sarma M.N.** entitled Complexity Theoretic Aspects of Rank, Rigidity and Circuit Evaluation may be accepted as fulfilling the dissertation requirement for the Degree of Doctor of Philosophy.

____ **Date :**

Chairman : V. Arvind

____ **Date :**

Convener : Meena Mahajan

____ **Date :**

Member 1: N.S. Narayanaswamy

____ **Date :**

Member 2: K.V. Subrahmaniam

____ **Date :**

Member 3: C.R. Subramanian

____ **Date :**

Final approval and acceptance of this dissertation is contingent upon the candidate's submission of the final copies of the dissertation to HBNI.

I hereby certify that I have read this dissertation prepared under my direction and recommend that it may be accepted as fulfilling the dissertation requirement.

____ **Date :**

Guide : Meena Mahajan

DECLARATION

I, hereby declare that the investigation presented in the thesis has been carried out by me. The work is original and the work has not been submitted earlier as a whole or in part for a degree/diploma at this or any other Institution or University.

Jayalal Sarma M.N.

ACKNOWLEDGEMENTS

One of my favourite quotes reads *To do what you like is freedom, To like what you do is happiness*. Thanks to many, I was free and happy, during the phase of my life when this thesis work was done. I will mention a few who made invaluable contributions that I am aware of, towards this venture.

It is my pleasure to thank my advisor Meena Mahajan. Right from my first day at IMSc, I was fortunate enough to receive guidance and encouragement from her in all academic endeavours that I was into. She has been a great teacher and a friendly and accessible mentor. As my research advisor, she gave me the freedom to choose my own research problems, and came up with timely guidance, ideas and support especially at difficult phases of this work. Her professional, systematic and down-to-earth approach to research, clarity of writing and presentation, are among the many things that I always proudly try to imitate.

Many results that appear in this thesis have been obtained in collaborative efforts. I thank my collaborators : Abhinav Kumar, Nutan Limaye, Satya Lokam, Meena Mahajan and Vijay Patankar for their contributions to this thesis in terms of ideas and insights. In addition, I thank V. Arvind, N.S. Narayanaswamy and C. R. Subramanian for being active members in my doctoral committee, and providing their inputs and support during the regular committee meetings. Thanks to K.V. Subrahmanyam for many insightful discussions we had on some topics of this thesis.

I have been fortunate to have an opportunity to work closely with Satya (Lokam) and Vijay (Patankar). I learned a lot from their wealth of experience, in terms of mathematical ideas and approach to research life. Thank you, for all the conversations we had, not excluding the academic discussions, ranging from topics like - when not to submit a paper to a conference, to - which restaurent to go to, for dinner. I express my gratitude to both of them, and Microsoft Research India, for facilitating my frequent visits to the Microsoft Research Lab in Bangalore, and for the kind hopitality that they extended during those visits.

My coursework at IMSc (and CMI) has been extremely useful in building up my outlook towards theoretical computer science and mathematics. I thank R. Balasubramanian, Kamal Lodaya, Meena Mahajan, K.N. Raghavan, Venkatesh Raman, R. Ramanujam and K.V. Subrahmanyam for the wonderful courses they offered during my first year at IMSc/CMI. Thanks to Kamal Lodaya for several non-technical educating conversations. I also thank K. Muralikrishnan and Vinod Pathari for their guidance during my undergraduate years, and for encouraging me to take academic research as a career route.

I thank The Institute of Mathematical Sciences for their generous funding for my doctoral studies in the form of fellowships, national and international travel grants, and contingency grants. The calm atmostphere in the institute has been extremely motivating for academic work. I thank all the staffs of IMSc, for the amount of work they put in to facilitate smooth functioning of my academic endeavours.

I have been blessed with many good friends at IMSc, who have always been there during difficult times for me to reflect upon myself. Special thanks are due to Ved (Prakash Gupta), Piyush (Kurur), Muthu(kumar T.), Narayanan N., Raghavendra (Rao), and Sunil (Easaw Simon).

Last, but perhaps the most, I would like to thank my parents and my extended family, for supporting my choices all along, especially when I wanted to take the road that is usually not taken, after my undergraduate studies that had many things to offer. This doctoral thesis, the first of its kind from the family, is indeed dedicated to them.

Abstract

This thesis studies some combinatorial, topological and linear algebraic parameters associated with Boolean and Arithmetic circuits. It is mainly divided into two parts.

The first part describes a study of combinations of graph-theoretic or circuit-theoretic restrictions that we can impose on Boolean circuits to obtain complexity-theoretic characterisations for the circuit value problem (CVP). We first address the question of evaluating monotone planar circuits (MPCVP). Using recent insights developed in the context of topological constraints in small-width circuits, we - in this thesis - review the developments leading up to and beyond the "MPCVP is in NC " result, and make some improvements on the known bounds for general MPCVP as well as some special cases. Our main improvements are obtained while considering circuits with cylindrical embeddings. Another contribution is that we are able to extend the NC upper bound on MPCVP to toroidal (genus one) monotone circuits.

Exploring how topological restrictions interfere with those in circuit theoretic parameters, we show that unless $\mathsf{P} = \mathsf{NC}$ in the non-uniform setting, there are P-computable functions requiring super-polylogarithmic number of negation gates in any poly-sized planar circuit computing them. In order to achieve this we prove that any circuit C with poly-logarithmic number of negation gates can be evaluated in NC. In a similar spirit, we prove an NC upper bound for evaluating a circuit which has poly-logarithmic crossing height when presented along with an embedding which achieves this crossing height. Combining these results, we show that any circuit C which has at most polylog crossing number and use polylog number of negations can be evaluated in NC when presented with along with an embedding which achieves this crossing height.

Motivated by applications in circuit complexity bounds, in the second part of the thesis we study the complexity of some linear algebraic parameters associated with the circuits. We first explore the circuit and computational complexity of matrix rank. This problem, in general is known to characterise a complexity class inside P. We study several restricted cases of the problem to obtain algebraic characterisations of the complexity classes. For instance we prove that computing the rank, over $\mathbb{Q}$, of matrices that are symmetric, non-negative and diagonally dominant, exactly characterises deterministic $\log$-space computation by Turing machines..

We next turn to optimisation problems associated with matrix rank. and briefly survey the applications of these problems in proving lower bounds in circuit complexity theory. Motivated by these applications we study the complexity of computing the rigidity of a

matrix : the minimum number of entries of the matrix that need to be changed in order to bring down the rank below a given value. We consider several variants of the problem, and characterise them in terms of complexity classes. In particular, we prove complexity theoretic characterisations for the problem when restricted to 0-1 matrices, and k is bounded by a constant. We also note that, in general, over $\mathbb{F}_2$, approximating the minimum number of changes needed up to a constant factor is NP-hard. We then consider the bounded norm variant of the problem, where changed matrix entries can differ from the original entries by at most a pre-specified amount. We note that it is NP-hard to compute this too.

We next attempt to construct explicit matrices which have super-linear rigidity. In this setting, we formulate the problem using the language of algebraic geometry, and prove tight maximal bounds for a specific family of matrices over $\mathbb{C}$. We then study continuity properties of matrix rigidity function, and prove that rigidity function is not semi-conituous in general, but for some special families of matrices, there is semi-continuity property. In the setting of the lower bounds, we apply and extend some known combintorial techniques to show almost optimal lower and upper bounds that for rigidity of a restricted triangular matrices.

Contents

List of Figures

List of Tables

List of Publications/Reports

[1] Nutan Limaye, Meena Mahajan, and Jayalal M.N. Sarma. Evaluating Monotone Circuits on Cylinders, Planes and Tori. In *Proceedings of 23rd International Symposium on Theoretical Science(STACS)*, volume 3884 of *Lecture Notes in Computer Science*, pages 660–671, February 2006. Journal Version to appear in *Computational Complexity* under the title "Improved Upper Bounds for Monotone Circuit Value: Some Restrictions and Generalisations".

[2] Meena Mahajan and Jayalal M.N. Sarma. On the Thickness of Branching Programs. Presented at Workshop on Computational Complexity and Decidability in Algebra (WCCDA 2007), Ekaterinburg, Russia, September 2007.

[3] Meena Mahajan and Jayalal M.N. Sarma. A Note on Evaluating Crossing Limited circuits. Manuscript, June 2007.

[4] Meena Mahajan and Jayalal M.N. Sarma. On the Complexity of Matrix Rank and Rigidity. In *Proceedings of 2nd International Computer Science Symposium in Russia (CSR)*, volume 4649 of *Lecture Notes in Computer Science*, pages 269–280, September 2007. Journal version to appear in the special issue of *Theory of Computing Systems*.

[5] Meena Mahajan and Jayalal M.N. Sarma. Rigidity of a Simple Extended Lower Triangular Matrix. To appear in Information Processing Letters, February 2008. http://dx.doi.org/10.1016/j.ipl.2008.02.010.

[6] Abhinav Kumar, Satyanarayana V. Lokam, Vijay Patankar, and Jayalal M.N. Sarma. Using Elimination Theory to Construct Rigid Matrices. Manuscript, April 2008.

Chapter 1

Introduction

The Hilbert's program, formulated by German mathematician David Hilbert in the 1920s, aimed at formalising all existing theories to a finite, complete set of axioms, and provide a proof that these axioms were consistent. Attempts in this direction led to the formal definition of the notion of computation. Church and Turing proposed different formal ways to abstract them, based on intuitions from mathematics and physics. The robustness of these models, supported by physical intuition, led to the formulation of the *Church-Turing thesis*, which states that computational problems that are algorithmically solvable are those that can be solved using a Turing machine. Taking this philosophical view point further, the area of recursion theory addresses questions about functions that are computable in these models.

When physical electronic computers were constructed in the late 1940s, the importance of having algorithms which uses minimum resources became prominent. However, despite lot of efforts by various researchers, certain problems seemed to require enormous amount of computational resources for solving instances of reasonable size. These considerations led to a formalised scaled down version of recursion theory, called *complexity theory* that studies the power and limitations of efficient computation in various computational models in terms of different computational resources. Not surprisingly, since its inception, the area has received a lot of attention from both theoretical and practical point of view, as it proved exceedingly relevant to both. The rapid development of this area of research in the last four decades established many surprising connections to various aspects in physical, mathematical, biological sciences. In particular, the area has its special interplay with various branches of mathematics such as combinatorics, graph theory, linear algebra, geometry, analysis etc., which played vital roles in settling many questions in this area.

Needless to say, despite these efforts, many important questions still remain open.

Circuits and Computation

Despite its simplicity, the model of Turing machines is not fully amenable to algebraic and combinatorial analysis. Historically, although developed independently, circuits as a model of computation proved to be a useful abstraction to bridge the gap. They are simply directed acyclic graphs, with a designated set of nodes called the inputs, and the rest are called gates, among which a few are called outputs.

The history of theoretical investigations about circuits goes back to the days of Shannon and Lypanov [Sha49, Lyp58]. Shannon proved close connections between what was then called switching circuits and Boolean algebra, and thus to the notion of circuits in the modern terminology. The initial stage of these developments was quite independent from the classical Turing machine based complexity theory. The fundamental question that was considered in those contexts was of the flavour : *how many gates does a switching circuit need exactly to compute a particular function?*. Thus, the number of gates in the circuit, or the size of the circuit, forms a resource under consideration. Other resources considered, in the context of circuits, are depth of the circuit (the graph), the width of the circuit, and the fanin of each gates (the in-degree of each node). Various attempts to answer the most general form of these questions have provided many insights into the study of other associated mathematical structures. In a more practical development, the model also turned out to be a useful abstraction of the notion of parallel computation and communication.

An important characteristic of circuits, as a model of computation, is that the abstraction allows to use different circuits for different input lengths, as opposed to Turing machines where the same machine description should work for all inputs. However, not surprisingly, under computational constraints (called *uniformity*,see [BI97]) on how to obtain the description of the circuit which works for all inputs of a given length, there are strong connections between circuit complexity classes and Turing machine complexity classes. The investigation in this thesis revolves around this aspect, studying many problems relevant for both models of computation, under various resources. A classical example for this connection is that the class of problems computed by polynomial time Turing machines can also be computed by polynomial sized uniform circuits.

This connection is more tightly woven, when the resource under consideration is space; the number of cells on the working tape used by the Turing machine. This makes it more relevant in many practical applications, since space, as a resource, faithfully abstracts out the number of memory cells used by a particular algorithm when implemented on a physical computer. Two classical examples of these tight relationships are: non-deterministic Turing machines which use only $\log$ space, poly time with has access to a poly bounded stack solve exactly same problems that can be solved by a uniform family of Boolean circuits of $\log$ depth and poly size, bounded fanin $\wedge$ gates, and unbounded fanin $\vee$ gates [Ven91]. The class of problems solved by circuit of $\log$ width circuits of bounded fanin are exactly characterised by deterministic Turing machines which uses $\log$ space [Pip79]. There are also trade-off results of the form: the class of problems solved by a uniform family of circuits of polynomial size and constant width is characterised by family of circuits of $\log$ depth, poly size and constant fanin [Bar89].

Various other parameters of the circuit can also be considered as a computational resource. Some natural choices for these are topological parameters on the graph and the type of gates that the Boolean circuit can use. The motivation for studying these, again, is the hope of gaining more combinatorial insights into the structure of the class of problems considered above. This line of study has been initiated long back, and several constrained families of circuits have been shown to be as powerful as the general ones. In this thesis we revisit some of these problems and obtain new results.

Evaluating Restricted Circuits

Given a Boolean circuit C over n inputs $x_1, \ldots, x_n$, and an assignment $x_i = a_i$ for each variable x_i, the Circuit Value Problem (CVP) is to determine the value $C(a_1, \ldots, a_n)$. This is a fundamental problem in complexity theory, since circuits capture computation in a very natural and universal way. When each gate is labelled $\wedge$, $\vee$ or $\neg$, CVP is complete for the complexity class P. It remains complete if the circuits are monotone (no $\neg$ gates except at the leaves); it also remains complete if the underlying graph has a planar embedding. This raises an important question; what are the combinations of graph-theoretic or circuit-theoretic restrictions that we can impose and obtain complexity-theoretic characterisations. This thesis explores this question in four tracks.

Evaluating Monotone Planar Circuits: Goldschlager [Gol80] proved a striking result that if the circuit is simultaneously monotone and planar (MPCVP) and is in a certain normal form, then evaluating it is in NC . That is, it can be solved using polylogarithmic time, using polynomial number of processors. Subsequently, Dymond and Cook [DC89] improved the upper bound for this special case to LogCFL, and Kosaraju [Kos90] extended the result by showing that a less restrictive special case, namely that of layered upward planar monotone circuits (subsuming Goldschlager's case), is also in NC, in fact in NC^3. Independently and in parallel, Delcher and Kosaraju [DK95] and Yang [Yan91] showed that MPCVP in its full generality is in NC^4 and in NC^3 respectively. More recently, Barrington *et al*[BLMS99] showed that for monotone upward stratified circuits — the special case considered in [Gol80, DC89] — there is in fact an upper bound of LogDCFL. Recall that $\mathsf{L} \subseteq \mathsf{NL} \subseteq \mathsf{LogCFL}$, $\mathsf{L} \subseteq \mathsf{LogDCFL} \subseteq \mathsf{LogCFL}$, and $\mathsf{LogCFL} = \mathsf{SAC}^1 \subseteq \mathsf{AC}^1 \subseteq \mathsf{NC}^2$ (see appendix A for more details on basic complexity classes).

Using recent insights [BLMS99, HMV06, Han04, ADR05a], in the context of topological constraints in small-width circuits, we - in this thesis - review the developments leading up to and beyond the "MPCVP is in NC " result, and make some improvements on the known bounds for general MPCVP as well as some special cases. Our main improvements are obtained while considering circuits with cylindrical embeddings. Such embeddings strictly subsume upward planar embeddings, but are not strong enough to capture all of planarity. Another major contribution is to *extend the NC upper bound on MPCVP to toroidal (genus one) monotone circuits.* This result appears in Chapter 3.

Evaluating Negation-Limited Planar Circuits: Markov [Mar58] showed that to compute a Boolean function on n variables, $\lceil \log(n+1) \rceil$ negation gates are necessary and sufficient. Fischer [Fis74] showed that the same holds even when restricted to polynomial sized circuits. In contrast, Santha and Wilson [SW91] proved that there are functions requiring super-logarithmic number of negation gates in any poly-sized constant-depth circuit computing them. We show a conditional topological analogue of this result, restricted to P-computable functions : *unless* $\mathsf{P} = \mathsf{NC}$ *in the non-uniform setting, there are* P*-computable functions requiring super-polylogarithmic number of negation gates in any poly-sized planar circuit computing them.* In order to achieve this we prove that any circuit C with polylogarithmic number of negation gates can be evaluated in NC.

Evaluating Crossing-Limited Monotone Circuits: The crossing number of a path of a graph in a given (combinatorial) embedding is the total number of crossings in the embedding of the path. The crossing height of a vertex v of a directed acyclic graph G, with respect to an embedding of G, is the smallest integer h such that any path starting from v to a leaf has crossing number at most h. The crossing height of a circuit C with respect to an embedding, is the crossing height of the root gate with respect to that embedding. In a spirit similar to limiting the negations that the circuit can use, we prove an NC upper bound for evaluating a circuit which has poly-logarithmic crossing height when presented along with an embedding which achieves this crossing height. However, this does not imply a conditional lower bound as above, since computing the embedding which achieves this crossing height is hard in general.

Evaluating Crossing-Limited Negation-Limited Circuits: A natural combination of the above two considerations (in fact a stronger form of the result on negation limited circuits) gives the following: Any circuit C which has at most polylog crossing number and use polylog number of negations can be evaluated in NC when presented with along with an embedding which achieves this crossing height.

Evaluating Thickness-Limited Skew Circuits: Skew circuits are polynomial sized circuits where $\wedge$ gates is allowed to have exactly one input other that the inputs to the circuit. They exactly correspond to what are called *branching programs* where computation may be stated as a graph graph reachability problem. The $\wedge$ gate will decide the presence or absence of an edges, and the circuit evaluation essentially is equivalent to testing reachability between two designated nodes. It is quite clear that polynomial-size branching programs decide exactly the languages in NL and while counting paths in such branching programs characterises the corresponding counting complexity classes. A surprising result of Barrington [Bar89] establishes that all of $\log$ depth bounded fanin circuits can be captured by bounded-width branching programs (in fact, width 5 suffices).

Recently, there has been some work on the topological restrictions of the underlying graphs in the context of width bounded circuits. Hansen [Han04] proved that constant width planar circuits capture exactly constant depth circuits equipped with $\mod$ gates too. More recently, Allender et al. [ADR05a] extended this result, by showing that constant width circuits with polylog genus can also be simulated by constant depth circuits with the help of $\mod$ gates.

It is natural to ask similar questions in the case of branching programs too. In this direction, Barrington et al. [BLMS97] showed that constant width planar branching programs capture exactly the languages accepted by constant depth circuits. In this context, Hansen [Han04] proved that bounded width branching programs which can be embedded on a cylinder can be computed by constant depth circuits with the help of $\mod$ gates.

We explore this thread further. In particular, we concentrate on another generalisation of the planarity criterion, namely *thickness* of the circuit. This has been already considered in [ADR05a] adopting a non-standard definition of thickness. We clean up the literature in this direction a bit, and compare with the standard definitions of thickness of circuits.

Along similar lines, Roy [Roy06] proved a thickness characterisation of deterministic $\log$ space computation. We tighten these results and align them with the standard notions of thickness. We obtain tighter characterisations of NC^1, using some variations of the existing constructions. We found that even with thickness 2, the class of NC^1 can be completely captured. For another variant of thickness, namely *book thickness*, a thickness of 3 suffices. These results also scale up to deterministic $\log$ space computation and slightly improve some of the bounds that are obtained in [ADR05a]. We conclude that the thickness parameter is not fine enough to bring out the fine structure of NC^1.

Circuit Lower Bounds

Circuit complexity theory came into limelight in the early eighties when many researchers had the view that the model might be at the right level of abstraction for attacking an important (*non-uniform*) version of the, by then famous P vs. NP problem [Coo03]. Attempts to prove lower bounds for various parameters of the circuit family computing explicit functions were the drive for a decade since then. Many important separation results followed suit [Hås86, Raz87, Smo87, Raz88]. But more importantly, many insightful proof techniques were developed.

Failures to prove super-polynomial size lower bounds for explicit Boolean functions motivated researchers to get to a meta level of the proofs themselves, and prove theorems about the proof techniques. Such an attempt had already appeared in the Turing machine complexity setting, which proved that only *non-relativizing* [BGS75] proof techniques can answer some of the big questions in complexity theory. The *lower bound drive* during the 80s did provide many non-relativizing techniques which resulted in some weak separation results in circuit complexity, but were unable to attack the big questions.

Getting to a meta level again, Razborov and Rudich [RR94], introduced the concept of *natural proofs* and ruled out many techniques from being useful in proving strong separation theorems in circuit complexity. Their arguments crucially used the notion of pseudorandom generators, in order to derive consequences at the meta level. But in the last decade, many proof techniques have been discovered which overcome both the above barriers [Vin05, San07]. Although they were sufficient to prove the above meta level arguments to be inadequate, they do not seem to provide a way forward towards the big questions. The reason, again were discovered recently, by Aaronson and Wigderson [AW08], who came up with a new barrier through the notion of *algebraic relativisation*, which they proved that all current techniques satisfy. In other words, any proof that separates P from

NP will have to be non-relativizing in an algebraic sense. However, it is quite unclear how these meta level arguments compare with each other.

In a positive move, researchers also tried out proving lower bounds in the setting of arithmetic circuits. Arithmetic circuit complexity also forms a part of a more general framework of *algebraic complexity theory* [BCS97]. The arithmetic circuit model is similar to Boolean circuits except that the gates can compute more general arithmetic functions. In many respects, this poses the problems in a more mathematical domain, thus making it possible to use well developed mathematical techniques. A prominent example of this is the ongoing program called *geometric complexity theory* [MS07], which attempts to prove super-polynomial lower bounds for permanent functions using techniques from algebraic geometry and representation theory.

With a different perspective, instead of moving to the setting of arithmetic complexity as such, researchers also tried to capture the combinatorial notion of computation using algebraic problems. This turned out to be a promising area of research. Much of the work in this direction has been done with the view that more methods from algebraic structures could be used to obtain results about the power of the computational model in the Boolean setting itself. In order to facilitate such approaches to major questions in space bounded boolean complexity theory, which we will be interested in, it will be useful to have algebraic problems capturing space bounded complexity classes. We attempt to do this for some of the classes in this thesis.

Circuit Complexity of Matrix Rank

A series of seminal papers by a variety of people including Grigoriev, Chistov, Mulmuley, Valiant, Toda and Vinay [Gri76, Chi85, Mul87, Val92, Tod91, Vin91] set the stage for studying the complexity of computing matrix properties (in particular, determinant and rank) in terms of $\log$ space computation and poly-size polylog depth circuits. This area has been active for many years, and efficient parallel algorithms (NC upper bound) are known for many related problems in linear algebra; see for instance [All04]. Some of the major results in this area are that computing the determinant and checking singularity of integer matrices characterise important complexity classes. In addition, the complexity of computing the rank of a given matrix over $\mathbb{Q}$ has been well studied. For general matrices, checking if the rank is at most r is has been characterised in a complexity theory perspective [ABO96].

The problem has also been studied under various restricted cases. An important combinatorial object associated with a matrix is the graph that it represents. Hence constraints on this graph, is a natural way to impose restrictions on the matrix. In this direction, Braverman et.al [BKR07], studies the circuit complexity of rank computation of restricted matrices where the matrix entries are constrained by imposing constraints on the graph which it represents. However, it is important to note that we do not know any combinatorial parameter on the graph which characterises the rank of its adjacency matrix. Braverman et.al. also studied the circuit complexity of matrix rank under restrictions on the field in which the rank is computed.

Taking a different route, we impose constraints that are more algebraic in nature. More precisely, we consider restrictions which are combinations of non-negativity, 0-1 entries, symmetry, diagonal dominance, tridiagonal and diagonal support, and we consider the complexities of three problems: computing the rank, computing the determinant and testing singularity. One of the interesting results that we obtained in this direction is that *computing the rank, over $\mathbb{Q}$, of matrices that are symmetric, non-negative and diagonally dominant, exactly characterises deterministic* $\log$*-space computation by Turing machines.*

Optimising Matrix Rank

Optimisation search problems for the above matrix properties can be considerably harder. In particular, an optimisation version of the rank computation problem would ask, given a matrix M and an integer r, what is the matrix *nearest* to M which has rank at most r. The notion of *nearness* is important in this consideration. The problem was first studied under the 2-norm (see Chapter 6 for a definition by Eckart et.al. [EY36]). Several variants of this problem were considered since then [Rum03a, Rum03b], and connections with the well studied notion of *condition numbers* were discovered .

We consider a variant of the problem in Chapter 8. Over any field, computing rank is known to be in NC [Chi85, Mul87]. Now consider the following existential search question: Given a matrix M over a field $\mathbb{K}$, a target rank r and a bound k, decide whether the rank of M can be brought down to below r by changing at most k entries of M. In Chapter 8, we consider several variants of the problem, and characterise them in terms of complexity classes. In particular, we prove complexity theoretic characterisations for the problem when restricted to 0-1 matrices, and k is bounded by a constant. We also note that, in general, over $\mathbb{F}_2$, approximating the minimum number of changes needed up to a constant factor is NP-hard.

We also consider the bounded norm variant of the problem, where changed matrix entries can differ from the original entries by at most a pre-specified amount θ. This variant behaves very differently. For a given a matrix M, a rank r and a bound θ, it may not be possible to bring the rank down to r even if you allow changing all the n^2 entries. Using results from [Roh89], we show that it is NP-hard to even test this.

Matrix Rigidity

Now we come back to the circuit lower bounds setting. As discussed above an important direction in which the attempts for proving lower bounds for circuit parameters have been partially successful is in the case of arithmetic circuits. We now come to a linear algebraic concept which is related to proving super linear lower bounds on circuit size.

For an $n \times n$ matrix over any field, the rigidity function $R_M(r)$ is the minimum number of entries that need to be changed to bring down the rank of the matrix below r. A folklore result is that over any field, $rank(M) - r \leq R_M(r+1) \leq (n-r)^2$. This concept, defined in [Val77, Gri76], is directly related to the rank optimisation question that we described above. The rigidity of a matrix is the smallest value of k for which the answer is affirmative.

An important result in this direction, established by Valiant [Val77], says that if for some $\epsilon > 0$ there exists a $\delta > 0$ such that an $n \times n$ matrix M_n has rigidity $R_{M_n}(\epsilon n) \geq n^{1+\delta}$ over a field $\mathbb{F}$, then the transformation $x \rightarrow Mx$ cannot be computed by linear size logarithmic depth linear circuits. See [Che05] for a survey of this result. Razborov [Raz89] (Lokam [Lok95]) proved that good lower bounds on rigidity (bounded norm variant) over a finite field (over reals) imply strong separation results in communication complexity. : For an explicit infinite sequence of (0,1)-matrices $\{M_n\}$ over a finite field F, if $R_M(r) \geq \frac{n^2}{2^{(\log r)^{o(1)}}}$ for some $r \geq 2^{(\log\log n)^{\omega(1)}}$, then there is an explicit language $L_M \notin \mathsf{PH}^{cc}$, where PH^{cc} is the analog of PH in the communication complexity setting. See chapter 6 for a precise statements and brief survey of these results.

However, obtaining explicit bounds on the rigidity of explicit family of matrices is surprisingly elusive, and thus has received a lot of attention (see introduction of [Lok95] and Chapter 6 for a survey). Lokam [Lok00] observed combinatorial limitations of the known approaches towards proving lower bounds for matrix rigidity. More recently, Lokam [Lok06] proved an unconditional quadratic lower bound for rigidity for a specific family of matrices (over $\mathbb{C}$). However, similar results are not known for $\mathbb{Q}$ or for finite fields $\mathbb{F}_q$ for any $q \geq 2$.

We provide a different way to overcome the combinatorial barrier, using tools from algebraic geometry. Our approach is simple, first we consider the dimension of the space of rigid matrices (for particular n, r and k). We formulate the problem in the language of algebraic geometry. We then use theorems from elimination theory to show existence of certain polynomials which have to be satisfied by matrices of rigidity k. This approach enabled us to prove tight bounds; for a specific family of matrices (again over $\mathbb{C}$) we could prove that the rigidity *is exactly* $(n-r)^2$. Although this does not lead to an asymptotic improvement over the results in [Lok06], we believe that this new technique can be used to prove lower bounds for other families of matrices.

Having a matrix over $\mathbb{C}$, to obtain a matrix over $\mathbb{Q}$ with the same rigidity, the natural approach is to turn to analytical properties of rigidity as a function over $\mathbb{C}$. In Chapter 7, we asked if the rigidity function is *lower semi-continuous* (i.e., does the value of the function drop suddenly in the neighbourhood of a point, with respect to the underlying topology). The hope is that we might be able to produce explicit rigid matrices over $\mathbb{Q}$ taking "good" rational approximations of the entries of the matrix produced in [Lok06]. However, we found that the answer is negative in general. However, using the framework of elimination theory, we argue that for some special families of matrices, we can take rational approximations.

The rareness of matching, or even close, lower and upper bounds correlates well with the lack of upper bounds on the computational version of rigidity. Due to the difficulty in obtaining non-trivial bounds, the exploration of combinatorial techniques that may lead to such bounds becomes interesting. A rare case where a closed-form expression has been obtained for rigidity is full-1s lower triangular matrices ([PV91]). We apply and extend their techniques to full-1s extended lower triangular (*elt*) matrices. In an elt matrix, the first diagonal above the main diagonal can be non-zero, but all other elements above the diagonal must be 0. We show lower and upper bounds that differ by an additive factor of roughly n/r.

Structure of the thesis

This thesis is divided into two parts. The first part addresses the effect of topological restrictions on the circuit, on the complexity of evaluating them. In Chapter 2, we present the improved upper bounds for monotone circuit value problem. In Chapter 3, we present extensions of these techniques to the case of higher genus circuits, circuits with limited negations, circuit with limited crossing number and finally circuits with limitations on crossing number and negations. In Chapter 4 we address the effect of topological restrictions on branching programs.

The second part of the thesis deals with algebraic parameters associated with circuits.

The circuit complexity of rank computation is discussed in Chapter 5. The notion of optimising matrix rank is introduced in Chapter 6. This chapter is a very brief survey of the various applications of matrix rigidity in complexity theory, and of the previous attempts to prove lower bounds in matrix rigidity. In the end of this chapter 6 we present the result on almost tight rigidity bounds for extended lower triangular matrices.

We get to the lower bounds in Chapter 7 where we present the algebraic geometric formulation of the rigidity problem, dimension bounds of the space of rigid matrices, and application of elimination theory to obtain negative conditions on rigid matrices, and finally choosing the entries of the matrix such that it fails satisfy these conditions, and hence remains rigid. In chapter 8 of the thesis, we present the complexity results concerning the problem of computing the rigidity of a matrix.

In three appendices (A,B and C) we present the basic material from complexity theory, algebraic geometry, and algebraic number theory that will be needed in this thesis.

Part I

Topological Constraints in Boolean Circuits

Chapter 2

Monotone Planar Circuit Value Problem

The P-complete Circuit Value Problem CVP, when restricted to monotone planar circuits MPCVP, is known to be in NC^3, and for the special case of upward stratified circuits, it is known to be in LogDCFL. In this chapter we re-examine the complexity of MPCVP, with special attention to circuits with cylindrical embeddings. We characterise cylindricality, which is stronger than planarity but strictly generalises upward planarity, and make the characterisation partially constructive. We use this construction, and three key reduction lemmas, to obtain several improvements. We show that stratified cylindrical monotone circuits can be evaluated in LogDCFL, and arbitrary cylindrical monotone circuits can be evaluated in AC^1(LogDCFL), while monotone circuits with one-input-face planar embeddings can be evaluated in LogCFL. For monotone circuits with focused embeddings, we show an upper bound of AC^1(LogDCFL). We re-examine the NC^3 algorithm for general MPCVP, and note that it is in $\mathsf{AC}^1(\mathsf{LogCFL}) = \mathsf{SAC}^2$. For formal definitions of the complexity classes in this chapter, see the appendix. The results in this chapter appears in [1].

2.1 Introduction

Given a Boolean circuit C over n inputs $x_1, \ldots, x_n$, and an assignment $x_i = a_i$ for each variable x_i, the Circuit Value Problem CVP is to determine the value $C(a_1, \ldots, a_n)$. This is a fundamental problem in complexity theory, since circuits capture computation in a very natural and universal way. When each gate is labelled AND, OR or NOT, CVP is complete for the complexity class P. It remains complete if the circuits are monotone (no NOT gates); it also remains complete if the underlying graph has a planar embedding.

However, if the circuit is simultaneously monotone and planar (MPCVP), then evaluating it is in NC.

The history of MPCVP begins with the papers of Goldschlager, where it is shown that planar CVP and monotone CVP are P-complete [Gol77], and that a special case of MPCVP, upward stratified circuits (see Section 2.2 for a formal definition) is in NC^2 [Gol80]. Subsequently, Dymond and Cook [DC89] improved the upper bound for this special case to LogCFL, and Kosaraju [Kos90] extended the result by showing that a less restrictive special case, namely that of layered upward planar monotone circuits (subsuming Goldschlager's case), is also in NC, in fact in NC^3. Independently and in parallel, Delcher and Kosaraju [DK95] and Yang [Yan91] showed that MPCVP in its full generality is in NC^4 and in NC^3 respectively. More recently, Barrington, Lu, Milterson and Skyum [BLMS99] showed that for monotone upward stratified circuits — the special case considered in [Gol80, DC89] — there is in fact an upper bound of LogDCFL.

There has recently been a spurt of activity examining topological constraints in small-width circuits [BLMS99, HMV06, Han04, ADR05b]. These works provide more insights into how to exploit the restricted topology. Using these insights, we review the developments leading up to and beyond the "MPCVP is in NC" result, and make some improvements on the known bounds for general MPCVP as well as some special cases. (However, we do not consider width restrictions in this work.) Our main improvements are obtained while considering circuits with cylindrical embeddings. Such embeddings strictly subsume upward planar embeddings, but are not strong enough to capture all of planarity. They have been studied in depth in the context of small-width circuits in [HMV06, Han04, Han08].

A key limiting problem that arises in our constructions is that of finding the length of a longest path in a planar directed acyclic graph (planar DAG). We define PDLP to be the class of problems log-space many-one reducible to this problem. While finding longest paths in general is hard, finding longest paths in DAGs is easily seen to be in NL, and in fact, NL-complete. It is conceivable, however, that the longest path problem over planar DAGs is considerably easier than NL. Recently, in [LMN08], this problem was shown to be in $\mathsf{UL} \cap \mathsf{coUL}$. However, since there are no completeness results known for PDLP, when we need longest paths in planar DAGs, we state our upper bounds explicitly in terms of PDLP rather than UL, keeping in mind that $\mathsf{PDLP} \subseteq \mathsf{UL} \cap \mathsf{coUL}$.

The main contributions in this chapter are as follows:

1. We characterise cylindrical graphs as spanning subgraphs of single-source single-sink

planar DAGs (Theorem 2.4). This is implicit in the result of Hansen (Theorem 2 of [Han04]), where layered cylindrical graphs are characterised as subgraphs of single-source single-sink layered planar DAGs. We state it explicitly because we obtain a partial logspace-constructive version, even when the given DAG is not layered to begin with. (Layering, in general, could be harder than logspace.) These results are presented in Section 2.3.

2. We present (in section 2.4) three reduction lemmas (Lemma 2.7, 2.8, and 2.9) which are at the heart of the improvements we obtain. The topological constraints considered are shown in Figure 2.1. The thick arrows go from stronger to weaker constraints, the dotted arrows indicate logspace reductions, and the dashed arrows indicate the reductions in $\mathsf{L}(\mathsf{PDLP})$. Using the reduction lemmas, in section 2.5, we obtain improved upper bounds for various version of the problem; see Table 2.1 [1]

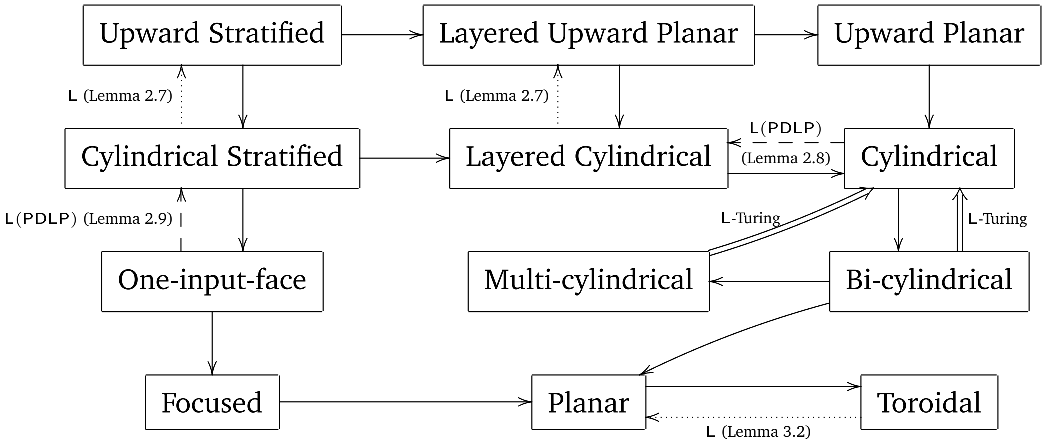


Figure 2.1: Relationship between various topological restrictions in the context of MCVP

2.2 Basic definitions

2.2.1 Circuits

A circuit C with n inputs $x_1, \ldots, x_n$ of size s is simply a directed acyclic graph on s vertices, with the vertices assigned one of the following types: (1) vertices whose in-degree is 0 are

[1] Some of the results in Table 2.1 are proved in Chapter 3, but we include them here for completeness.

(Monotone) Circuit type	Embedding	Our upper bound	Previous bound
Cylindrical stratified	given	$\mathsf{LogDCFL}$ (Thm. 2.10)	NC^2 ([Yan91] Sec. 2)
One input face	not needed	$\mathsf{L}(\mathsf{PDLP} \oplus \mathsf{LogDCFL})$ (Thm. 2.11) $\subseteq \mathsf{LogCFL}$	NC^2 ([Yan91] Sec. 3)
Cylindrical	given	$\mathsf{AC}^1(\mathsf{LogDCFL})$ (Thm. 2.14)	–
Planar	not needed	$\mathsf{AC}^1(\mathsf{LogCFL}) = \mathsf{SAC}^2$ (Thm. 2.17)	NC^3 [Yan91]

Table 2.1: Improved upper bounds

called the input nodes and are assigned one of the literals or 0 or 1, (2) vertices whose in-degree and out-degree are non-zero are called gates, and (3) the vertices whose out-degree is zero are called output nodes.

We consider circuits with gates labelled AND, OR, NO-OP. A gate labelled AND or OR has fan-in two, a gate labelled NO-OP has fan-in one, and a gate labelled by a constant has fan-in zero and is a source node. Without loss of generality, we assume that constant gates have fan-out one and that no gate has fan-out greater than two. We do not assume that there is a single sink [2].

A circuit with variables is a circuit in which some fan-in zero gates are labelled by variables. By generalised circuits we mean circuits which also have constant gates with non-zero fan-in and possible fan-out more than one; the output of such a gate is independent of its inputs, but the input wires could play a role in determining the planar embeddings. Generalised circuits, with or without variables, arise in the recursive steps of the algorithms from [DK95, Yan91].

A circuit is said to be *layered* if there is a partition $V = V_0 \cup V_1 \cup \ldots \cup V_h$ such that all edges go from some layer V_i to the next layer V_{i+1}. A circuit is said to be *stratified* if it is layered and all source nodes are in layer V_0.

A language L is said to be in NC if there is a family of polynomial-size polylog depth circuits $\{C_n\}$ with AND, OR, and NOT gates, with all NOT gates at the leaves, such that $x \in L$ iff $C_{|x|}(x) = 1$. Circuit C_n having depth $O(\log^i n)$ corresponds to NC^i if the AND/OR gates have bounded fan-in, to AC^i if they have unbounded fan-in, and to SAC^i if only the AND gates are constrained to bounded fan-in. Clearly, $\mathsf{NC}^i \subseteq \mathsf{SAC}^i \subseteq \mathsf{AC}^i \subseteq \mathsf{NC}^{i+1}$.

[2] The earlier NC algorithms for MPCVP made this assumption, since if there are multiple sinks, each of them can be evaluated independently. However, removing nodes with no path to the designated sink may not be possible in logspace, so we explicitly note this as a computational requirement.

2.2.2 Topological Embeddings and Drawings

In this paper, we are concerned with directed acyclic graphs, denoted DAGs. Though many of the definitions below apply to general graphs, we will use them specialised to DAGs.

A graph is said to be *planar* if it can be embedded in the plane without crossings. That is, the nodes and edges of the graph can be drawn in such a way that the representations of no two edges intersect, except at shared endpoints. A plane graph is a graph along with a planar embedding. Note that planarity is independent of whether the graph is directed or not. By the results of [RR94, AM04, Rei05], deciding if a given graph is planar and if so finding a planar embedding is in AC^1, SL , and now L.

A planar embedding is *bimodal* if at every vertex v, all outgoing (incoming) edges appear consecutively around v. It is easy to see ([TT86], [Han04] Lemma 5, [Yan91] Lemmas 3.1 and 3.2) that in a planar DAG with a single source and a single sink, (a) every embedding is bimodal, and (b) for every face f, the edges incident on f form a simple (undirected) cycle consisting of two directed paths.

A planar embedding of a DAG is said to be a *one-input-face embedding* if all source nodes lie on the same face. Testing if a planar DAG is one-input-face, and if so, uncovering such an embedding, is easy: add a new source node with edges to all the old sources, and test for planarity.

A drawing (not necessarily planar) of a digraph on the plane is *upward* if the drawing of every edge is monotonically increasing in the vertical direction. Every DAG has an upward embedding, which can be recovered by a topological sort. (Also, only DAGs have upward embeddings, since a cycle cannot be embedded in an upward way.)

A digraph is *upward planar* if it has an embedding that is simultaneously upward and planar. Though all DAGs are upward, not all planar DAGs are upward planar. Figure 2.2 shows a standard instance of a planar DAG which is not upward planar (see for instance [BT88]). In fact, given a planar DAG, deciding whether it is upward planar is NP-complete [GT01]. (It is also known that every upward planar graph has an upward planar embedding using only straight-line drawings of all edges [BT88]. Furthermore, if the DAG is layered, all nodes in the same layer will have the same y-coordinate.)

A digraph is *cylindrical* if it can be embedded on a cylinder surface, in a way such that all edges are monotonically increasing in the direction of the axis of the cylinder. (Clearly such a digraph must also be acyclic, a DAG.) As observed in [Han04], this generalises upwardness, with the edges embedded on the surface of the cylinder rather than on a plane. Note that the surface of the cylinder can be embedded on a plane in a straightforward

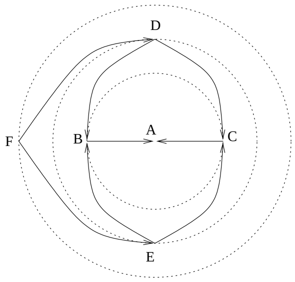


Figure 2.2: A planar DAG that is cylindrical but not upward planar

way: place the right end of the cylinder (the end towards which all edges flow) on the plane, and dilate the cylinder in a continuous way into a cone section until its surface lies flat around the end placed first. (In fact, the converse is also true: any embedding on the plane can be drawn on the surface of the cylinder. But the edges may not be monotone along the cylinder axis.) Thus a cylindrical embedding will give rise to a planar embedding where all edges flow in an inward direction towards a central face. It follows that every cylindrical embedding is also bimodal, even if it is not single-source single-sink.

Cylindricality strictly generalises upward planarity, as Figure 2.2 shows. The example of Figure 2.3 shows that cylindricality does not capture all planar DAGs.

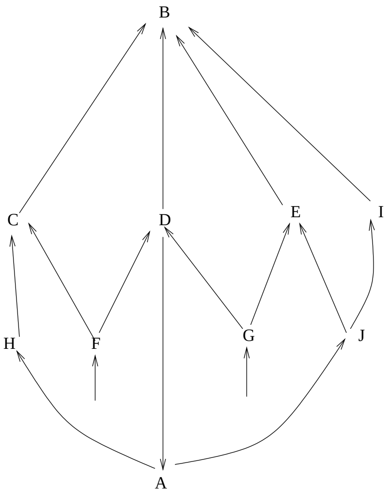


Figure 2.3: A planar DAG that is not cylindrical

A *layered cylindrical embedding* of a layered digraph is a cylindrical embedding where layers correspond to disjoint circles of the cylinder (or concentric circles on the plane, in the corresponding inward drawing). In recent literature in the graph drawing community, the term *radial drawing* is used. For instance, the radial levelled planar drawings of [BBF03] are exactly layered cylindrical embeddings. We continue to use the term cylindrical rather than radial, since the main issue in radial levelled planar drawings appears to be: given the partition of the vertex set into sets lying on the same layer, find the ordering on each layer. On the other hand, we are often concerned with finding the partition as well, and this could well be a harder problem.

Recall that a layered circuit (in general, a layered DAG) is said to be stratified if all source nodes appear at layer 0. A DAG is said to be *upward stratified* (*cylindrical stratified*) if it is layered, stratified, and has an upward planar (cylindrical respectively) embedding. It follows that an upward/cylindrical stratified circuit has a one-input-face embedding. Figure 2.4 shows a layered planar DAG which has an upward planar embedding and a one-input-face embedding but no upward one-input-face embedding. In [DK95], the term *restricted stratified* is used to denote circuits which are *cylindrical stratified* as defined above (without the *restricted*, the authors of [DK95] mean *generalised* circuits). On the other hand, in [BLMS99], *stratified* refers to *upward stratified* as described here.

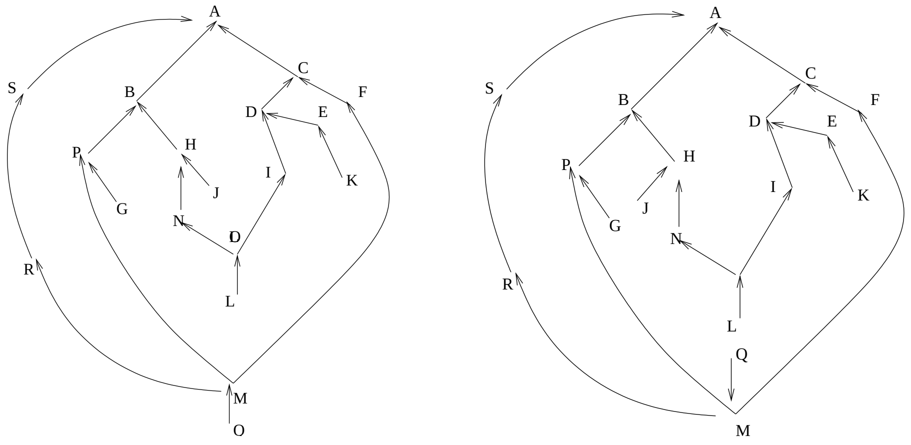


Figure 2.4: A layered planar DAG with an upward planar embedding and a one-input-face embedding but with no upward one-input-face embedding

A planar embedding of a DAG G is *focused* if there is a subset S of source nodes, all of which are embedded on a single face, and every node of G not reachable from S is itself a source node. This is a topological analogue of a skewness condition on circuits. Note that one-input-face embeddings are (vacuously) focused; S is the set of all source nodes.

We use the terms SSPD and SMPD to mean single-source single-sink planar DAGS and single-source multiple-sink or multiple-source single-sink planar DAGs respectively.

2.2.3 Representing embeddings

Planar embeddings: By the results of [RR94, AM04, Rei05], deciding if a given graph is planar and if so finding a planar embedding is in AC^1, SL , and now L. The embedding so obtained is a planar combinatorial embedding, specifying the cyclic (clockwise, say) ordering of edges around each vertex in some plane embedding. (In fact, specifying for each vertex the clockwise cyclic ordering of edges around it is what is called a combinatorial embedding, and corresponds to an embedding of the graph on some orientable surface of appropriate genus.) Checking whether a given combinatorial embedding corresponds to an embedding on the plane can be done in logspace (see [AM04]).

We briefly discuss how faces are specified in any planar embedding. Recall that embeddings ignore directions on edges. In fact, for each (undirected) edge (u, v), the embedding will specify where arc (u, v) figures in the circular list around u, and where arc (v, u) figures in the circular list around v. The arcs (u, v) and (v, u) are expected to be superimposed in the corresponding geometric embedding. We use the term edges to refer to directed edges of the original graph, while we use the term arcs to refer to the directed arcs in the combinatorial embedding. For every arc $e = (u, v)$, there are faces $L(e)$ and $R(e)$ to the left and right, respectively, of the edge. (These could both be the same, if, say, e is a bridge in the underlying graph.) If G is a connected graph when directions on edges are ignored, then for every face f, the set of edges e with $f \in \{L(e), R(e)\}$ form a connected graph. This set can be traversed systematically as follows. Start with an arc $e = (u, v)$ such that, say, $f = R(e)$. Let $e' = (v, w)$ be the arc preceding (v, u) in the cyclic ordering around v. Then $f = R(e')$. Keep advancing in this way until the starting arc is encountered again; in the process, the entire boundary of f will be traversed. We assume that f is "named" by the lexicographically smallest arc $a = (u, v)$ such that $f = R(a)$. See [MT01, Whi73] for more about representing embeddings.

Layered cylindrical or Layered upward planar embeddings: We assume that the embedding is given in the following form: (a) the cyclic ordering of edges around each vertex (the planar combinatorial embedding) corresponding to the geometric embedding, and (b) the circular or left-to-right ordering of vertices at each layer. It is straightforward to see that given such information, we can verify in logspace that it indeed corresponds to some layered cylindrical or layered upward planar geometric embedding.

Cylindrical embeddings: For cylindrical embeddings of non-layered graphs, we need to specify some more information. Imagine circles drawn along the surface of the cylinder, through each vertex. The ordering of the circles along the axis of the cylinder imposes a partial order on the vertices (total, if no two vertices lie on the same circle); consider any total order extending this. This ordering corresponds to non-decreasing distance of vertices from the bottom end of the cylinder. For each vertex u, we can talk of its left face and its right face: the left face is the face between u's leftmost incoming edge (last incoming edge in clockwise ordering) and leftmost outgoing edge (first outgoing edge in clockwise ordering), while the right face is the face between its rightmost incoming and outgoing edges. If u is a source, then the left and the right face are the same, and it is the face containing the (initial segment of) the ray drawn out of u against the cylinder axis. Similarly, if u is a sink, it is the face containing the (initial segment of) the ray drawn out of u along the cylinder axis. Given the clockwise ordering of edges around each vertex, the left and right faces can be determined for each u that is not a source or sink. For a source/sink u, if we explicitly specify the leftmost outgoing/incoming edge, then this face can be determined. We call this edge $L(u)$. For instance, see the example in Figure 2.5. The total order is $A\ B\ E\ F\ C\ D$. For source A, $L(A) = (A, B)$, while for sink D, $L(D) = (C, D)$. The left faces of B and C are f_l and f_r respectively. The right face of B is the region inside the quadrilateral $BFEA$, while the right face of C is the region inside the triangle BCD.

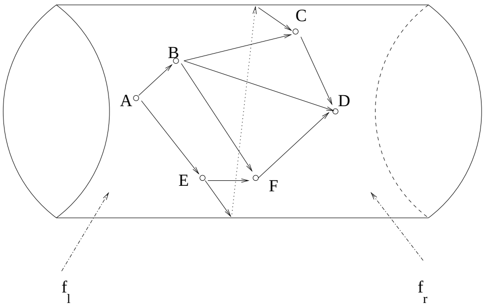


Figure 2.5: Representing a cylindrical embedding

With this background, we now assume that the following information about the cylindrical embedding is available: (a) the cyclic ordering of edges around each vertex (the planar combinatorial embedding), (b) a total order $v_1, v_2, \ldots, v_n$ of the vertices, extending the partial order induced by the cylindrical embedding, and (c) for each source/sink u, the edge $L(u)$. In particular, the edges $L(v_1)$ and $L(v_n)$ specify the faces f_b and f_t corresponding to the bottom and top ends of the cylinder.

It is not clear that given (a),(b),(c) above, one can check in logspace if the corresponding plane embedding is cylindrical. However, this information is sufficient for the results of this paper.

2.3 Graphs on cylinders

Upward planar graphs have been characterised independently in [Kel87] and [BT88]: A DAG is upward planar if and only if it is a spanning subgraph of a planar st-digraph, that is, a planar DAG with a single source s, a single sink t, and an edge from s to t. Extending this result, [Han04] characterises layered cylindricality: a layered digraph is layered cylindrical if and only if it is a subgraph of a layered planar DAG with a unique source and a unique sink (an SSPD). While the result is implicit in the work of [TT89], the major contribution in the proof of [Han04] is to make the transformation uniform. In a similar vein, we characterise cylindricality (without the layered property); while the topological ideas are already there in the proofs of [TT89, Han04], we prove it in a different way to obtain suitable uniformity bounds. We then use these to evaluate cylindrical circuits.

One direction of our characterisation crucially uses a layered embedding algorithm independently due to [Yan91] and [DK95]. The algorithm of [Yan91] is stated for single-sink digraphs where there is a one-input-face planar embedding (an embedding in which all sources appear on the same face), while that of [DK95] is stated for what are called focused circuits. We will use the algorithm for single-sink one-input-face planar DAGs, and we observe that this includes, as a special case, SSPDs. ([Yan91] uses the notation layered one-input-face for cylindrical stratified (all source nodes at the first layer)). An important property of such embeddings is that all vertices are bimodal; thus left and right faces of a vertex are defined. The algorithm is described in Figure 2.6.

Steps 1-2 of the algorithm provide the layering, step 3 provides the cylindrical embedding of the layered graph. To see why the algorithm is correct, see Section 3 of [Yan91] or [DK95]. We observe the following:

Proposition 2.1. *The layered embedding algorithm above runs in* L(PDLP). □

Now we establish our characterisation by the following two lemmas.

Input: a one-input-face single-sink planar directed acyclic graph H.

Output: A layered cylindrical embedding of a graph H', obtained from H by subdividing edges into directed paths.

Method: Let t be the sink of H.

1. For each node v in H find the longest distance $d(v)$ to t. Let $d = \max_v\{d(v)\}$; there are $d+1$ layers. The input nodes are in V_0. A non-input node u is in layer $l(u) = d - d(u)$.
2. For a directed edge (u, v) in the graph, let $k = l(v)-l(u)-1$. If $k > 0$, then introduce dummy nodes $n_1, n_2 \dots n_k$ and add the edges (u, n_1),$(n_1, n_2) \dots (n_k, v)$. (That is, we subdivide edge (u, v) into a directed path of length $l(v) - l(u)$.) The dummy node n_i will be in layer $l(u) + i$.
3. For each node u (including dummy nodes), walk along the boundary of the left (or right, respectively) face of u beginning at u. The first node encountered with the same layer number as u is the left (or right, respectively) neighbour of u.

Figure 2.6: Layered embedding algorithm ([Yan91] Section 3, [DK95] Section 4)

Lemma 2.2. *If a planar DAG G is a spanning subgraph of an SSPD H (a planar DAG with a single source and a single sink), then G has a cylindrical embedding which, given G and H, can be constructed in* L(PDLP).

Proof. Using the algorithm of Fig. 2.6, a cylindrical embedding can be found for H' obtained from H by edge subdivision. Replacing the directed paths obtained through subdivision by original edges, we get a cylindrical embedding of H, and hence of G. The upper bound for constructing the embedding of H follows from Proposition 2.1. □

Lemma 2.3. *If a planar DAG G has a cylindrical embedding, then it is a spanning subgraph of a cylindrical DAG H with a single source and a single sink.*

Proof. Consider the layout of the graph on the cylinder surface, with vertices in order $v_1, v_2, \dots, v_n$ as specified by the cylindrical embedding. Clearly, v_1 is a source and v_n is a sink. Without loss of generality, we assume that the circles of the cylinder through v_1 and v_n do not contain any other vertex. (If they do, move vertex v_1 slightly towards the cylinder bottom, v_n towards the top. This does not change the combinatorial specification of the embedding.)

If any vertex v_i other than v_n is a sink, we need to add an edge from it to some v_j with $j \geq i$ without destroying cylindricality. Such a v_j can always be found as follows: imagine a particle moving out of v_i along the direction of the cylinder axis. It aims to avoid intersecting any edge. So if it encounters an edge, it moves parallel to and infinitesimally close to the edge. Since all edges are cylindrical, its movement is still monotonic with respect to the axis. As soon as it reaches (infinitesimally close to) a vertex, we declare that vertex to be v_j. If it never encounters an edge or a vertex, then it will exit at the right end of the cylinder. In this case we declare v_n to be the desired v_j. The movement of the particle ensures that the edge (v_i, v_j) can be added preserving cylindricality. A similar procedure applied after this will work to make all sources other than v_1 have incoming edges. □

Theorem 2.4. *Let G be any planar directed acyclic graph. The following are equivalent.*

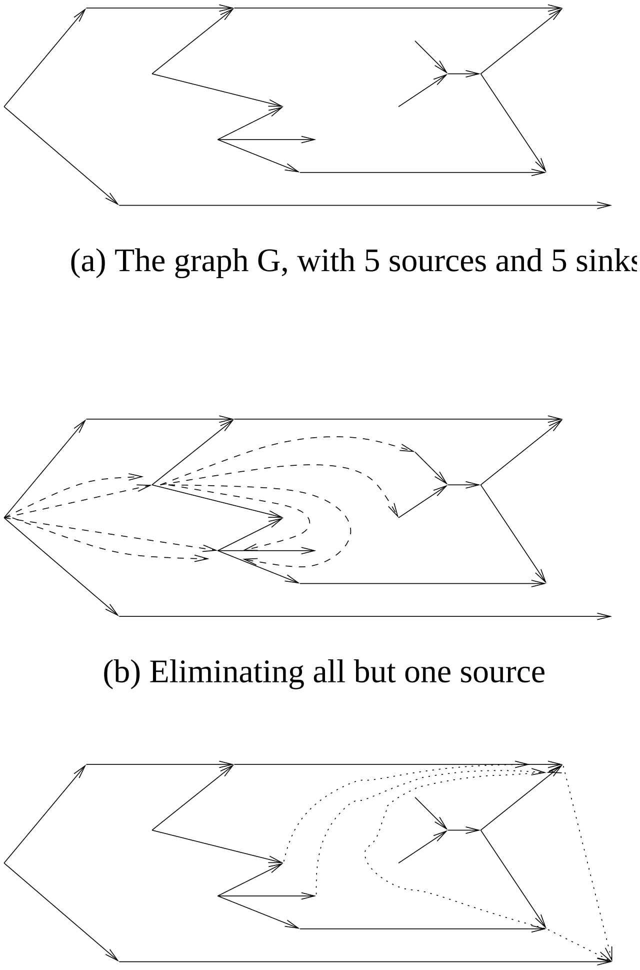
1. *G has a cylindrical embedding.*
2. *G is a spanning subgraph of a cylindrical SSPD.*
3. *G is a spanning subgraph of an SSPD.*

It follows that testing for cylindricality is in NP. However, though cylindricality generalises upward planarity, testing for which is NP-complete, it is possible that testing for cylindricality is easier.

One direction of the theorem above is already constructive using Lemma 2.2. We make the proof of Lemma 2.3 constructive via a more complicated construction. This construction works only for one stage (multiple sinks to single sink or multiple sources to single source), and yields only a planar (not cylindrical) embedding of H. The advantage is that it is implementable in logspace.

Lemma 2.5. *Let G be a connected (in the undirected sense) cylindrical DAG with S sources and T sinks. Given the cylindrical embedding of G, we can construct, in L, a planar single-source DAG H_s with T sinks and a planar single-sink DAG H_t with S sources such that G is a spanning subgraph of both.*

Proof. We describe how to construct H_s; the construction of H_t is symmetric. Since G is connected, for every face f, the edges incident on f form a connected graph. For each face f, let i be the smallest index such that v_i is on the boundary of the face. Then there is some edge $e = (v_i, v_j)$ such that $f = R(e)$. Start traversing the boundary of f, starting with such an edge $e = (v_i, v_j)$. For each v_k encountered on the boundary with in-degree 0, add edge (v_i, v_k). See Figure 2.7 (a), (b) for an example.



(a) The graph G, with 5 sources and 5 sinks

(b) Eliminating all but one source

(c) Eliminating all but one sink

Figure 2.7: Obtaining H from a connected G.

Clearly this preserves acyclicity, since all new edges are from a lower indexed to larger indexed vertex. This also preserves planarity. The new edges are inserted, in the order encountered, into the cyclic ordering around v_i immediately after the arc (v_i, v_j). A new edge (v_i, v_k) is inserted into the cyclic ordering around v_k immediately after the arc (v_l, v_k) which led to the discovery of v_k on this face boundary. Thus we can easily compute the new planar combinatorial embedding.

As the figure shows, we may end up adding far more edges than is necessary. (Multiple edges will not get added if we process each face sequentially. But in logspace, we cannot cascade polynomially many such stages. So while processing each face, we check for in-degree zero in the original graph.) Since G is connected, every source has an (undirected) path to v_1. Hence every source lies on the boundary of at least one face with a lower indexed vertex, and hence acquires an incoming edge. Thus at the end, only v_1 is a source. □

As figure 2.7 (b) shows, applying the above construction on a graph to remove multiple sources may trap a sink. So we cannot sequentially remove multiple sources and then multiple sinks. In fact, after removing multiple sources, we do not know if the graph H so constructed necessarily has a cylindrical embedding. Even if it does, we do not know how to recover one.

In the above proof, connectedness ensured that every source other than v_1 acquired an incoming edge. We observe in the following lemma that absolute connectedness is not a critical requirement.

Lemma 2.6. *Let G be a cylindrical DAG where each connected component of the underlying undirected graph has either a single source or a single sink. Then a planar single-source single-sink DAG H of which G is a spanning subgraph can be constructed in* L.

Proof. Partition the underlying undirected graph of G, in L, into connected components $G_1, \ldots, G_c$. For each component, there is a cylindrical embedding inherited from that of G, which can be efficiently retrieved. By Lemma 2.5, each G_i is a spanning subgraph of a planar DAG H_i, with a single-source s_i and single-sink t_i, and H_i can be constructed in logspace. All that remains is to combine these H_i. Since each H_i is acyclic, the graph H obtained by adding edges t_i, s_{i+1} is also acyclic, and has a single source s_1 and single sink t_c. To see why it is planar, consider planar embeddings of each H_i with s_i on the external face. (The construction of Lemma 2.5 does yield such embeddings.) Consider any face f for which t_i is on the boundary. We insert the embedding of H_{i+1} in this face, and connect t_i to s_{i+1}. (See Figure 2.8.)

To construct a planar embedding of H, we can simply construct afresh a planar embedding of H in L. (Strictly speaking, this is not necessary. The edge (t_i, s_{i+1}) can be inserted anywhere in the cyclic ordering of t_i. In the cyclic ordering of s_{i+1}, it should be inserted in such a way that it lies on the external face of H_{i+1}. Given the way H_{i+1} is constructed from G_{i+1}, this information about the external face is indeed available.) □

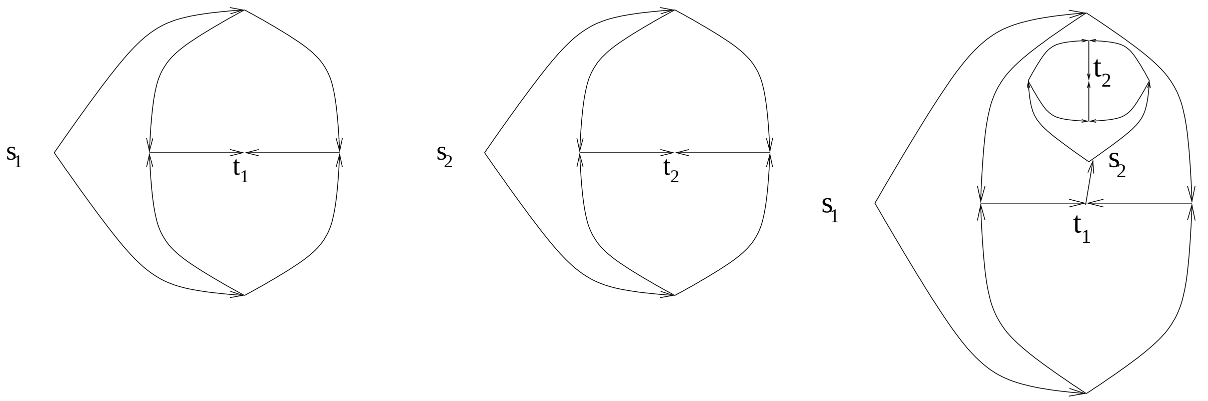


Figure 2.8: Patching H_1 and H_2 preserving planarity.

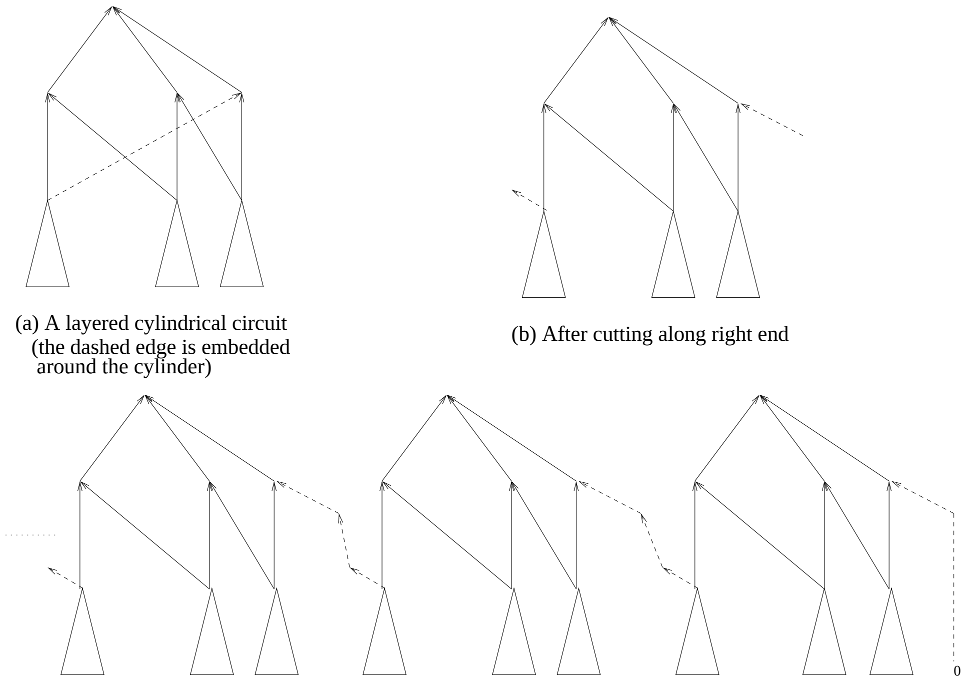
2.4 Circuits on cylinders

We now show that for circuit evaluation, any technique applicable to layered upward planar circuits also applies to cylindrical circuits, with a uniformity requirement in $\mathsf{L(PDLP)} \subseteq \mathsf{UL} \cap \mathsf{coUL}$. The result is obtained in two stages: first we show how to deal with layered cylindrical circuits, and then we show how to layer arbitrary cylindrical circuits. We also show that one-input-face circuits reduce to upward stratified circuits, with a similar uniformity requirement.

Lemma 2.7. *Given a circuit C with a layered cylindrical embedding $\mathcal{E}$, we can in logspace obtain an equivalent circuit C' with a layered upward planar embedding $\mathcal{E}$'. Further, if $\mathcal{E}$ is stratified, so is $\mathcal{E}$'. Also, if C is monotone, so is C'.*

Proof. Intuitively, what we want to do is as follows. Consider a geometric embedding of C on the plane, with layers corresponding to concentric circles and edges travelling inwards. By rotating a ray shooting out of the root, we can find an angular position where it does not contain the embedding of any node. By deforming edge representations if necessary, we can ensure that each edge intersects the ray (at this angular position) in at most one point. Now simply "cut" the circuit C along the ray. This gives rise to dangling in-edges and out-edges and a circuit D which is layered upward planar. Patch multiple copies of D side-by-side, feeding zeroes to the dangling edges of the extremal copies, and let the root of the middle copy be the new root. See Figure 2.9.

To translate this into a formal proof, we need to describe (a) how to obtain, in logspace, the curve along which we will cut the circuit C to get D, (b) how the copies will be patched functionally, (c) how the embeddings of the copies will be patched, and (d) why



(a) A layered cylindrical circuit (the dashed edge is embedded around the cylinder)

(b) After cutting along right end

(c) Joining up copies of cut circuit

Figure 2.9: Obtaining an upward circuit equivalent to a cylindrical one.

the resulting circuit is equivalent to C.

We first perform some preprocessing on the circuit. Since we are given the layering as well as the label r of the circuit output gate, we can throw away all gates at a larger layer than r. Now, treating all edges as undirected, use the logspace connectivity algorithm to delete all gates with no (undirected) path to r. Let the resulting circuit be C_1, with layers $V_0, V_1, \ldots, V_h$ and r at layer h. We replace each vertex u by vertices u_{in} and u_{out} with a directed edge from u_{in} to u_{out}. The type of gate u_{in} is the type of u, while u_{out} is a NO-OP gate. An edge (u, v) is replaced by the edge (u_{out}, v_{in}). The resulting circuit, call it C_2, has $2h$ layers: an out layer for V_0, an in layer for V_h, and two for all other layers. The layered cylindrical embedding of C_2 is easily obtained from that of C_1, and hence of C, in logspace. The only tricky point is handling sources/sinks of C_1. If u is a source of C_1, we need to decide where to insert the edge (u_{in}, u_{out}) into the cyclic ordering of edges leaving u_{out}. This is where we need the third part of the representation of cylindrical embeddings: we insert this edge just before the edge $L(u)$. Similarly for a sink v, we insert (v_{in}, v_{out}) in the ordering around v_{in} just after $L(v)$. C_2 is clearly equivalent to C; further, it has the nice property that no layer has a source as well as a sink.

To see (a), we start with vertex r of C_2. At some stage, suppose that the path ρ under construction has reached vertex g from above. If g is at the lowest layer, we are done. Otherwise, move down to any neighbour of g at a lower layer. Suppose there is no such neighbour; that is, g is a source node. Then g is of the form v_{in} for some $v \in C_1$. Traverse the boundary of the face to the right of (v_{in}, v_{out}), until it first encounters a vertex g' at a layer lower than g. Such a vertex must exist, since g has undirected connectivity to r which has undirected connectivity to the layer below g. The path ρ now proceeds from g to g'.

The path ρ constructed uses some circuit edges and some dummy edges. Let C_3 be the graph $C_2 \cup \rho$. The above procedure of constructing ρ also gives us a layered cylindrical embedding of C_3.

We cut C_3 to the immediate right of the path, starting at r, to obtain a layered upward planar circuit C_4. The embedding of C_4 is specified as follows: Retain edges (u, v) where neither u nor v is on ρ. For u on ρ, retain edges leaving or entering u to/from the path ρ or the left of ρ. Replace an edge (u, v) leaving ρ on its right by the edges (x, v) and (u, x'), where x and x' are new gates of fan-in/fan-out zero. Similarly, replace an edge (w, u) entering ρ from its right by the edges (y', u) and (w, y), where y' and y are new gates of fan-in/fan-out zero. It is clear that this can be performed in logspace. C_4 is the circuit D informally described earlier.

Let d be the depth of C_2. Place $2d + 1$ copies of C_4 side by side in a row. Identify new node x' of copy i with new node x of copy $i + 1$. Identifying x and x' gives a subdivision of an edge present in a copy of C_4. Restore the subdivision to a single edge (remove the identified node). New nodes x of the leftmost copy, and new nodes x' of the rightmost copy, are fed constant 0, via paths of NO-OP gates of appropriate length (this is done to preserve stratifiedness). See Figure 2.9 (c). Designate the root of copy $d + 1$ as the new root. Let this circuit be called D. It is easy to see that D is layered upward planar, and that its embedding can be obtained from that of C_4 in logspace. Also, if C_2 is stratified, so are C_4 and D.

We claim that D is equivalent to C_2, and hence to C. The reason is simple: at the lowest level, all nodes of D are correct (they evaluate to the same value as corresponding nodes in C_2). If at level l, the copies $i - 1, i, i + 1$ of C_4 are correct, then at level $l + 1$ the ith copy of C_4 is correct. Thus over $2d + 1$ levels, we may lose at most $2d$ copies, but the central copy will correctly evaluate the root of C_2. □

In the above proof, the layering of the given circuit appears crucial. We observe below that without layering, the same conversion can be performed in L(PDLP).

Lemma 2.8. *Evaluating a circuit C with a cylindrical embedding $\mathcal{E}$ reduces in* L(PDLP) *to evaluating a layered cylindrical circuit C' with embedding $\mathcal{E}$'. Further, if $\mathcal{E}$ is one-input-face, then $\mathcal{E}$' is stratified. Also, if C is monotone, so is C'.*

Proof. We proceed in four steps.

1. We remove from C all nodes with no directed path to the output gate of C. This gives an equivalent circuit G with a single sink, and with an inherited cylindrical embedding.

2. From the given cylindrical embedding of G, we construct the SSPD H with the same vertices as G and containing all the edges of G.

3. Using the layered embedding algorithm of Figure 2.6, we obtain a layered cylindrical embedding of an SSPD H', obtained by subdividing edges of H into directed paths.

4. We recover a layered cylindrical embedding of a digraph G' from that of H' by simply throwing away all directed paths corresponding to edges in $H \setminus G$. We convert G' to a circuit by specifying that all the new subdivision vertices have type NO-OP.

Since C is a planar DAG, Step (1) can be performed in L(PDLP). Step (2) uses Lemma 2.5, and can be performed in logspace. Step (3) uses Lemma 2.2, and runs in L(PDLP). It is straightforward to see that Step (4) can be performed in logspace. □

Note that the layered embedding algorithm needs a single-sink one-input-face embedding. In the above proof, the one-input-face condition is achieved in step 2 by exploiting cylindricality. However, if the given circuit already has a one-input-face embedding, then cylindricality is not needed. Thus we have:

Lemma 2.9. *Evaluating a circuit C with a one-input-face embedding $\mathcal{E}$ is reducible, in* L(PDLP)*, to evaluating a stratified cylindrical circuit C' with embedding $\mathcal{E}$'. Also, if C is monotone, so is C'.* □

2.5 Improved Upper bounds for MPCVP

In this section we revisit some of the MPCVP algorithms in the literature. We observe that some of these algorithms have tighter bounds than claimed. Wherever possible, we apply (some of) the reduction lemmas of Section 2.4 to expand the class of circuits for which the algorithm applies. Wherever possible, we also try to weaken the input requirements.

Goldschlager [Gol80] considered upward stratified circuits. He showed that in this special case, if the corresponding embedding is given with the input, then MPCVP is in NC^2. This upper bound was improved to LogCFL by Dymond and Cook [DC89]. They use the characterisation (due to [Coo71, Sud78]) of LogCFLas languages accepted by polynomial-time-bounded pushdown automata augmented with an auxiliary logspace worktape, AuxPDA(poly) in short. (Similarly, LogDCFL is characterised as languages accepted by deterministic polynomial-time-bounded pushdown automata augmented with an auxiliary logspace worktape, DAuxPDA(poly).)

The main idea behind obtaining the LogCFL bound is as follows: since the circuit is monotone, intervals of contiguous 1s at the input level travel upwards as contiguous segments which may shrink, expand, or merge, but never split. (This last property breaks down if the embedding is not stratified.) So evaluating the given circuit C amounts to proving that an interval is true (or valid), by finding a set of intervals at the previous level which imply its validity, and recursively proving their validity. An important property of a minimal set of intervals proving validity of the root (a "proof tree" on intervals) is that it is polynomial sized; hence an auxiliary push-down automaton performing the recursive verification nondeterministically will run in polynomial time. But this is precisely the class LogCFL.

The work of Barrington *et al.*[BLMS99] brings the evaluation of monotone upward stratified circuits, presented along with such an embedding, down to LogDCFL by evaluating the circuit in a bottom up fashion. The DAuxPDA algorithm repeatedly transforms the input by (a) detecting when a 0- or 1- interval at the input layer fails to propagate high enough, and (b) replacing the interval by all 1s or all 0s. The transformation thus preserves the value of the output gate. The stack is used to keep track of the frontier up to which simplifying transformations have been made. Polynomial running time is ensured, amongst other things, by the placement of a virtual blocking interval of 0s on either extreme at each level. The algorithm requires the upward stratified embedding to be supplied as input. Though not stated explicitly, it also works for circuits with multiple sinks. (The only point to be checked is that intervals of 1s may merge though separated not just by a 0 interval but by 0- and 1- intervals, all arising at sinks; see the discussion preceding Proposition 8 of [BLMS99]. This makes no difference to the technical claims.)

Since virtual blocking intervals cannot be placed at extremes of each layer for a cylindrical embedding, we do not see how to extend this algorithm to work for stratified cylindrical circuits. However, we can still obtain this upper bound by using Lemma 2.7 in conjunction with this algorithm:

Theorem 2.10. *Given a monotone planar circuit C with a stratified cylindrical embedding, determining whether C evaluates to 1 is in* LogDCFL. □

What if the embedding needed for Theorem 2.10 is not explicitly given, but there is the promise that such an embedding exists? At some cost, we can recover a suitable embedding. The cost is high enough that we can weaken the premise further. Note that stratified cylindrical embeddings are one-input-face, though the converse may not hold. But one-input-face embeddings can be constructed in logspace. With such an embedding, we can apply Lemma 2.9 and Theorem 2.10; thus we get a slightly weaker upper bound for a more general class:

Theorem 2.11. *Given a monotone planar circuit C, if C has a one-input-face embedding, then C can be evaluated in* $\mathsf{L}(\mathsf{PDLP} \oplus \mathsf{LogDCFL})$.

Proof. We first construct a one-input-face embedding of C in logspace, as described in Section 2.2.2. Then we apply Lemma 2.9 to obtain an equivalent cylindrical stratified circuit C', and use Theorem 2.10. □

Layered one-input-face circuits were considered by Yang [Yan91] as a step towards placing general MPCVP in NC . Note that these are precisely cylindrical stratified circuits. In Section 2 of [Yan91], an upper bound of NC^2 is obtained for evaluating such circuits. Rather than use a tool like Lemma 2.7 followed by the bound of [Gol80], Yang devised a somewhat different algorithm, since a modification of it was used in a later section. The essence of his algorithm was the same as in [DC89]: evaluating the given circuit C is equivalent to evaluating a circuit C' which tries to determine, for each interval or segment of gates at each level, whether this interval evaluates to all 1s. Further, he carried the range of inputs used in proving validity as a parameter. That is, for each interval i, j of gates numbered between i and j at level l, and for each input range x, y, determine if the interval i, j, l can be proved valid using only inputs from the range x, y. (Note: it is not claimed that all inputs in the range x, y are 1s, merely that 1s outside this range are not needed for proving validity.) By doing this, he was able to establish that C' has polynomial algebraic degree. Then he appealed to [MRK88] to obtain the NC^2 bound. However, it is now known that circuits of degree polynomial in circuit size can be evaluated in LogCFL [Ruz80, Ven91]. Thus we have,

Proposition 2.12. *The algorithm of Section 2 from [Yan91], for evaluating instances of* MPCVP *presented with cylindrical stratified embeddings, has a* LogCFL *implementation.* □

Another notable point is that though Yang assumed a single-sink circuit, his algorithm works also in the presence of multiple sinks.

This bound was independently obtained by Delcher and Kosaraju [DK95], who observed that the algorithm of [DC89], though presented only for upward stratified circuits, works also for the cylindrical stratified case. This is because even for such embeddings, the proving sub-circuit for validity of intervals has a tree structure which is polynomial-sized.

In [Kos90], the requirement that the circuit be stratified was dropped for the first time. The input is required to be a monotone layered upward planar circuit, with the witnessing embedding supplied. Dropping the stratified (one-input-face, for layered circuits) condition means that intervals of contiguous 1s can split due to the presence of an input node at an intermediate layer, and this makes all the preceding algorithms for upward-planar or cylindrical stratified circuits inapplicable. Kosaraju's idea is, however, quite simple and elegant: repeatedly split the circuit horizontally at a layer such that both pieces are between 1/4 and 3/4 of the entire circuit in size. Evaluate each piece recursively, replacing cut off wires by variables. (The details of the recursive splitting are a bit sketchy in [Kos90] but are supplied in full in [DK95] for the stratified case.)

But what does it mean to evaluate a circuit with variables? Due to monotonicity, if a gate evaluates to 1 (0) even when all variables are set to 0 (1, respectively), then the gate evaluates to 1 (0, respectively) for all settings of the variables. So by evaluating such a circuit on two settings — all variables 1, and all variables 0 — the gates can be partitioned into three sets: evaluating to 1, or 0, or depending on the input variables. Once the recursive evaluation is done, the bottom piece is entirely evaluated and the top piece has some variable gates. But now the values of all its variable inputs are known from the bottom piece, so this piece can be fully evaluated.

Clearly, the recursion depth is logarithmic, and the base case of recursion is a monotone upward stratified circuit with variables. As observed above, [Kos90] used the fact that the NC^2 bound of [Gol80] applies also in the presence of variables to obtain the three-part partition. Using this bound for the base case, [Kos90] reported an upper bound of NC^3.

It is worthwhile noting that at internal stages of the recursion, the circuits could become generalised; they could have constant gates with non-zero fan-in (e.g. an OR gate could get as inputs one 1 and one variable from the preceding level of recursion). So, to apply Goldschlager's algorithm to the base case, the constant gates with non-zero fan-in are explicitly removed. That is, to patch up the two pieces, only the sub-circuit induced by gates which depend on variables is considered.

It is also worthwhile noting that this algorithm is also insensitive to multiple sinks, since the strategy evaluates not just a designated sink but every gate in the circuit.

Kosaraju's upper bound can be tightened by noting that a log-recursion-depth algorithm, using the algorithm of [BLMS99] rather than [Gol80] for the base case, yields an implementation in $\mathsf{AC}^1(\mathsf{LogDCFL})$.

Proposition 2.13. *The algorithm of [Kos90], for evaluating instances of* MPCVP *presented with layered upward planar embeddings, has an* $\mathsf{AC}^1(\mathsf{LogDCFL})$ *implementation.* □

Further, the class of circuits for which this bound applies can be expanded to cylindrical circuits:

Theorem 2.14. *An instance of* MPCVP*, presented with a cylindrical embedding, can be solved in* $\mathsf{AC}^1(\mathsf{LogDCFL})$. □

Proof. Let C be the given circuit with a cylindrical embedding. Using Lemma 2.8, we obtain in $\mathsf{L}(\mathsf{PDLP}) \subseteq \mathsf{NL} \subseteq \mathsf{AC}^1$ an equivalent circuit C' with a layered cylindrical embedding $\mathcal{E}$. Applying Lemma 2.7 gives, in $\mathsf{L} \subseteq \mathsf{AC}^1$, an equivalent layered upward planar circuit C'', to which the preceding proposition can be applied. Note that for subcircuits evaluated at recursive steps, embeddings are inherited from $\mathcal{E}$. □

Bi-cylindrical Circuits : We now consider a generalisation of cylindrical circuits, which we call *bi-cylindrical* circuits. These strictly subsume cylindrical, while still lying within planar circuits.

Definition 2.15 (Bi-cylindrical circuits)**.** *A DAG or circuit G is bi-cylindrical if it has an embedding on the surface of the cylinder such that there is a circle C going around the cylinder surface, and all edges go towards C.*

Thus C splits G into two pieces (overlapping only on C) where each piece is cylindrical. (See Figure 2.10.)

Now each piece can be evaluated separately, and the the root gate can then be evaluated from its values in the two pieces. Depending on whether the pieces are layered or not, and whether they have one-input-face embeddings or not (if both do, then all inputs lie on the two extreme ends of the bi-cylinder), we have the following upper bounds:

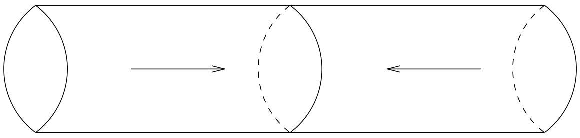


Figure 2.10: Bi-cylindrical embeddings

bi-cylindrical circuit type	layered	not layered
inputs only at extremes	LogDCFL	L(PDLP $\oplus$ LogDCFL)
inputs anywhere	AC^1(LogDCFL)	AC^1(LogDCFL)

Focused circuits : Focused embeddings are considered in [DK95], since they arise in recursive stages of their final algorithm for general MPCVP. Recall that a focused embedding is one where all sources other than those in a designated face f feed into a node reachable from a source in f. This is a topological analogue of a skewness condition on circuits. Such a circuit C can be converted to an equivalent upward stratified one C' (with such an embedding explicitly obtained) by simplifying the neighbours of the inputs not on the special face and then using Lemma 2.9 followed by Lemma 2.7. One consequence is that some internal nodes may be constant nodes; e.g. an OR gate with a skew 1 input from outside f simplifies to a constant gate, but still has another input wire feeding into it. We could cut off such wires as well. (But we must do this *after* obtaining the stratified cylindrical embedding; if we do it before that, then the resulting circuit is no longer one-input-face, so Lemma 2.9 does not apply.) After this cutting, the resulting circuit C'' will not be stratified, so we can only use the bound of Theorem 2.14 and not that of Theorem 2.10. Since C' can be obtained from C in L(PDLP) $\subseteq$ AC^1, and since C'' can be obtained from C' in logspace, we have:

Theorem 2.16. *Given a monotone planar circuit C with a focused embedding, determining whether C evaluates to 1 is in* AC^1(LogDCFL). □

The final algorithms of both [Yan91] and [DK95] make no assumptions about the embedding; given an instance of MPCVP with *any* planar embedding, they show that evaluation is in NC . Both algorithms repeatedly evaluate carefully chosen smaller circuits with special embeddings (cylindrical stratified or focused). But the noteworthy point is that these special embeddings for the smaller circuits can always be obtained, in NC , from the given planar embedding.

Yang's analysis proceeds by showing that $O(\log n)$ iterations of the following suffice: For each face f containing some inputs, consider the subcircuit C_f reachable (in a directed sense) from f. C_f can have some dangling in-edges from the rest of the circuit; replace these by variables to get a circuit with variables and a focused embedding. Evaluate this circuit as far as possible (the variables, or unknown wires, do not allow complete evaluation), using a generalisation of the scheme leading to Proposition 2.12. Then perform some obvious simplifications, and reiterate.

The generalisation does not permit the use of [BLMS99] or Theorem 2.10. However, the strategy is the same as originally used by Yang for one-input-face embeddings; namely, there is an equivalent polynomial degree circuit doing this partial evaluation. Hence, by [Ven91], it can be performed in LogCFL. Hence, a careful analysis of Yang's algorithm allows us to conclude that MPCVP is in $\mathsf{AC}^1(\mathsf{LogCFL})$. However, it can be seen that this class is the same as SAC^2. Thus we have the following:

Theorem 2.17. *Given a monotone planar circuit C, determining whether C evaluates to 1 is in* SAC^2. □

2.6 Discussion

This investigation leaves many questions unanswered.

1. Is cylindricality testing NP-hard? Recall that cylindricality strictly generalises upward planarity, testing for which is NP-hard ([GT01]), and is strictly stronger than planarity, testing for which is in L([RR94, AM04, Rei05]). Actually, upward planarity testing becomes hard only in the presence of multiple sources, but is in AC^1 for single-source planar DAGs [BBMT98].

2. How can a cylindrical embedding be represented so that given a representation of this form, verifying that it is indeed cylindrical can be done in logspace? The representation we have used does not seem to have enough information for this.

3. Given a graph with the promise that it is cylindrical/layered cylindrical/layered upward planar, what is the complexity of recovering a witnessing embedding? This can make a big difference to the complexity of circuit evaluation; see item 5 below.

4. Recently, via a different approach bypassing Figure 2.6, Theorem 2.11 has been improved: one-input-face MPCVP has been shown to be reducible to layered upward planar monotone circuits, and hence is in LogDCFL [CD06]. It appears that focused MPCVP can also be captured in LogDCFL via this approach.

5. There are very few hardness results with respect to topological constraints. A well-known result due to [Bus87] says that evaluating a Boolean formula (the circuit is a tree) is NC^1-complete, thus MPCVP is at least NC^1-hard. A more recent notable result [Han04] shows that constant-width planar circuits characterise ACC^0. Are there natural topological restrictions which, placed on MPCVP, give instances complete for LogDCFL, NL and LogCFL? In particular, is stratified cylindrical MPCVP hard for LogDCFL?

   In [CD06], one-input-face MPCVP is shown to be hard for L. The hard instance produced here is in fact a width-2 tree. However, the result of [Han04] does not imply that evaluating it is in ACC, because the ACC evaluation procedure of [Han04] explicitly needs the layered bounded-width presentation of the circuit, and it is computing this that is L-hard. Similarly, the result of [Bus87] does not imply that evaluating it is in NC^1, because the NC^1 evaluation procedure of [Bus87] requires the formula to be explicitly presented in fully parenthesised form, and computing this is L-hard. In other words, the hardness of evaluating one-input-face MPCVP lies in the hardness of obtaining a small-width specification, or even an explicit tree description, under the promise that the circuit is indeed a small-width tree. This situation thus underscores the difference that supplying an embedding can make; hence the importance of item 3.

   A special case of layered upward planar MPCVP arises when all AND gates are skew. (The hard instances of [CD06] are skew.) In this case, the circuit evaluates to 1 if and only if there is a path from an input labelled 1 to the root; it captures reachability in layered upward planar graphs. It is noteworthy that we do not know L-hardness for reachability in layered grid graphs, or even in grid graphs; the best lower bound is NC^1 (see [ABC$^+$06]). However, it is possible that layered upward planar monotone circuits are harder to evaluate than similar skew circuits.

6. Let DLP_i denote the class of problems logspace many-one reducible to the problem *DAGLONGPATH* where the DAGs are unrestricted for $i = 0$, planar for $i = 1$, planar single-source or planar single-sink for $i = 2$, and planar single-source single-sink for $i = 3$. (Thus, DLP_1 is what is referred to as PDLP till now in this paper.) Let DR_i denote the class of problems logspace many-one reducible to reachability in the corresponding DAGs. Clearly, $\mathsf{DR}_i \subseteq \mathsf{DLP}_i$, and $\mathsf{DLP}_0 = \mathsf{DR}_0 = \mathsf{NL}$. What other relationships can be deduced among these classes?

Notice that the layering algorithm of Figure 2.6 already needs a one-input-face single-sink planar DAG. A circuit on such a DAG can trivially be converted to an equivalent instance of DR_3 by adding a dummy source. Thus, the upper bounds of $\mathsf{L(PDLP)}$, obtained in Proposition 2.1 and Lemma 2.2, can actually be replaced by DLP_3, which may conceivably be stronger. In recent work by [ABC+06], DR_3 and DR_2 are shown to be in L. Thus, if DLP_3 can be shown to be equivalent to DR_3, or reducible to DR_2, then the upper bounds of this paper will drop further. We need to be a bit careful: Lemma 2.8, for instance, uses Proposition 2.1 as well as DR_1 (step 1 uses DR_1 to obtain an equivalent instance of DR_2), and thus has a fine upper bound of $\mathsf{L}(\mathsf{DLP}_3 \oplus \mathsf{DR}_1)$. To establish Lemma 2.9, on the other hand, $\mathsf{L}(\mathsf{DLP}_3)$ suffices, since the first step is also dispensable. These finer bounds can be carried over to all the results of Section 2.5.

Chapter 3

Extensions of Topological restrictions to other parameters

In this chapter we extend the ideas developed in the previous chapter to some non-planar cases, as well as to the non-monotone case. The results in this chapter appear in [1] and [3].

- We consider a generalisation to genus 1 (toroidal) in Section 3.1 and show that such monotone circuits can be evaluated in NC.
- We consider a restricted generalisation to higher genus in Section 3.2 and show that such monotone circuits can be evaluated in NC.
- We also consider planar non-monotone circuits with restrictions on the placement of negation gates, in Section 3.3, and show that such circuits too can be evaluated in NC.
- We consider monotone circuits with bounds on the crossing number of the circuit, in Section 3.4, and show that such circuits too can be evaluated in NC.
- We combine the above two restrictions and obtain NC upper bounds for circuits where the crossing number and the negations are bounded simultaneously.

See figure 3.1 and table 3.1 for listing and comparison of these restrictions with the ones in the previous chapter.

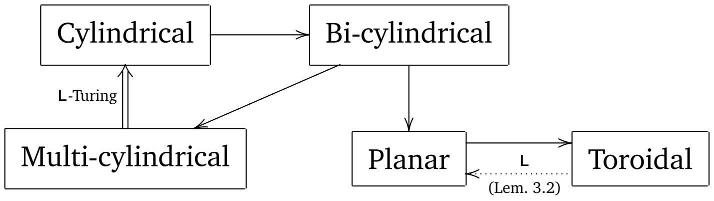


Figure 3.1: Relationship between various topological restrictions in the context of MCVP

Circuit type	Embedding	Our upper bound	Previous bound
Toroidal	given	SAC^2 (Thm. 3.3)	P
Multicylindrical	given	$\mathsf{AC}^1(\mathsf{LogCFL}) = \mathsf{SAC}^2$ (Table 3.2)	P P
Non-monotone Planar polylog negation-height	not needed	NC (Lem. 3.5,3.6)	P
Monotone, polylog Crossing Number	given	NC (Thm. 3.10)	P
polylog Negation Height polylog Crossing Number	given	NC (Thm. 3.12)	P

Table 3.1: Generalisations of MPCVP

3.1 Monotone Circuits on the Torus

We start with the case of a torus which is the canonical surface of genus 1. A digraph is *toroidal* if it can be embedded on a torus. We look at circuits whose underlying DAG is toroidal. We assume that the toroidal embedding is given as a combinatorial embedding; verifying that this embedding has genus one can be done in log space (see [AM04]).

Any closed curve separates the plane into disconnected regions, but a closed curve can disconnect the surface of a torus or leave it connected. In the latter case, it is called a surface non-separating curve. Any non-planar toroidal graph has at least one surface non-separating cycle. The following lemma is from [ADR05a]:

Lemma 3.1 ([ADR05a]). *Given a non-planar graph G with an embedding on the torus, a surface non-separating cycle in G can be found in* L.

Using this result, we establish the following reduction lemma, which along with Theorem 2.17, immediately gives the main result of this section.

Lemma 3.2. *A circuit C with a toroidal embedding can be converted in log space to an equivalent circuit C' with a planar embedding. Also, if C is monotone, so is C'.*

Proof. The lemma is proved by essentially using the idea from [ADR05a]. Intuitively what we want to do is as follows. Consider a given toroidal embedding. Using Lemma 3.1, we will find a cycle (in the undirected sense) such that "cutting" the circuit along the cycle will make the remaining graph planar. Now we will paste together several copies as in the cylindrical case (Lemma 2.7) such that one copy evaluates to the same function as the original circuit. Also, the pasting will be done preserving planarity.

As in Lemma 2.7, to translate this into a formal proof, we need to describe (a) how to obtain, in log space, the curve along which we will cut the circuit (b) how the embeddings of the copies will be patched, (c) how the copies will be patched functionally, and (d) why the resulting circuit is equivalent to C.

For (a) and (b), we use Lemma 3.1. Borrowing the notation from [ADR05a], let $v_1, v_2 \dots v_r$ be the non-separating cycle returned by the log-space procedure. Let G' be the graph obtained after cutting along this cycle. This graph will have two copies of the vertices on the cycle on each end of the cylinder. Let these be $v_{1,1}, v_{2,1}, \dots v_{r,1}$ and $v_{1,2}, v_{2,2}, \dots v_{r,2}$ respectively. Let d be the depth of the original circuit. We make $2d+1$ copies of the circuit and place them side by side, identifying the corresponding vertices and edges. The combinatorial embedding of C' is obtained exactly as in Section 3 of [ADR05a], see Figure 3.2 for an illustration. Clearly, C' is planar, since it has an embedding on the surface of the cylinder. (Note, however, that the embedding may not be "cylindrical".)

For (c), each gate in each copy behaves exactly as in the original circuit. Edges coming into the extreme copies from outside are set to source nodes with value 0. Let this new circuit be called C'.

Now to establish (d), we introduce the notion of *cycle-height*. Let c be the non-separating cycle with respect to which cutting has been performed. The cycle-height of gate g is the smallest non-negative integer k such that every path from a leaf to g "crosses" the cycle c at most k times. By a simple inductive argument, we can establish that if gate g has cycle-height k, then all copies of g in C', except those in the leftmost k and rightmost k copies of C, evaluate to the same value as g in C. It follows that in the middle copy, all the gates will get evaluated correctly. □

Theorem 3.3. *A monotone circuit, given with an embedding on a torus, can be evaluated in* SAC^2. □

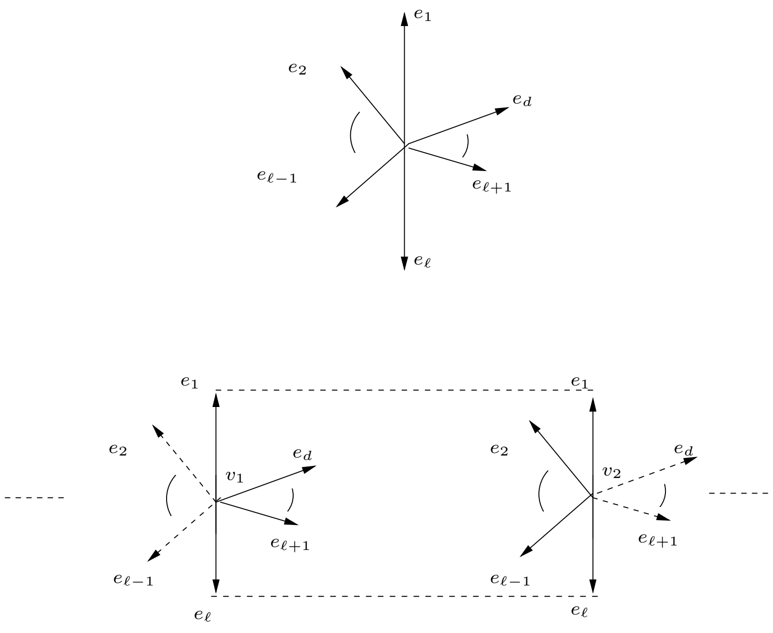


Figure 3.2: Patching the copies

An obvious question is whether the above technique can be extended to give an NC upper bound for higher genus circuits. The limitation is that if we do not get a genus 0 surface to make copies, then the process of making copies will increase the genus.

3.2 Monotone Multi-cylindrical circuits

We extend the idea of bi-cylindrical circuits in a natural way to what we call *multi-cylindrical* circuits. Such circuits strictly subsume the bi-cylindrical case, but are incomparable with planar circuits. A noteworthy point is that a multi-cylindrical circuit can be of arbitrary genus. The following definition captures this extension.

A *k-cylindrical* circuit can be presented as a set of k components. Each of these has a cylindrical embedding. The edges of each cylindrical component flow towards the right rim. And the right rims of each can be identified (let us call that curve c). Another circuit sits on the gates in c such that all the inputs to this circuit come only from gates in c. This circuit can be cylindrical stratified, cylindrical, planar or toroidal.

A *multi-cylindrical* circuit is a k-cylindrical circuit, for some k.

Notice that 2-cylindrical according to this definition is stronger than the bi-cylindricality discussed earlier. This is because we allow a circuit C' sitting on the nodes on c. But

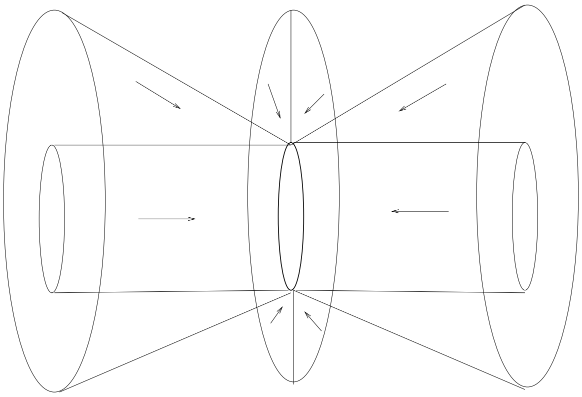


Figure 3.3: Multi-cylindrical embeddings

allowing this is also essential, since each gate is assumed to have fan-in at most 2. If such a construct were not allowed, then the root gate would itself have to sit on c and take inputs from at most 2 components. The other components would play no role at all and could be excised, making k-cylindrical equal to 2-cylindrical for $k > 2$. On the other hand, allowing such a construct, bi-cylindrical circuits are exactly those 2-cylindrical circuits for which C' is the trivial circuit; it merely pulls out the value of a fixed gate appearing on the curve on c.

Let C' be the subcircuit sitting over the nodes in c. Now C' can be thought of as a circuit which has c as its set of input nodes. We can evaluate each of the cylindrical components separately in parallel. With this, we get the value of each node in c. Now we can evaluate C' using the values of nodes in c. Depending on the complexity of evaluating each component, and of evaluating C' from c, we have the following upper bounds:

Inputs on c_i's	Type of C'	layered	not layered
only at extremes	cylindrical stratified	LogDCFL	L(PDLP $\oplus$ LogDCFL)
anywhere	cylindrical	AC^1(LogDCFL)	AC^1(LogDCFL)
anywhere	planar	AC^1(LogDCFL)	SAC^2
anywhere	toroidal	–	SAC^2

Table 3.2: Upper bounds for Multicylinderical Circuits

As one can see, this gives upper bounds only for the promise problem. Also, one limitation is that we do not know the complexity of obtaining such an embedding if one exists, and hence the embedding need to be explicitly given along with the input. As far as we know, this is the first result on evaluating a class of monotone circuits which contains some arbitrary genus circuits, in NC . Clearly, if P $\neq$ NC in the non-uniform setting, there are high genus circuits which do not have multi-cylindrical embeddings.

3.3 Circuits with Limited Negations

We now consider planar circuits which are not monotone, but where the negation gates are limited in some way. Without such a limitation, there is no hope of evaluating the circuit inside NC unless P= NC , since planar CVP is known to be P-complete [Gol77]. How many negation gates are needed to obtain this hardness? We show in this section that unless P= NC , there are P-computable functions requiring super-polylogarithmic number of negation gates in any poly-sized planar (and even toroidal) circuit computing them (Lemmas 3.5,3.6).

Markov [Mar58] came up with a surprisingly tight bound on the number of negation gates that are needed to compute any boolean function. He showed that to compute a boolean function on n variables, $\lceil \log(n+1) \rceil$ negation gates are necessary and sufficient. One natural question to ask is whether such a bound holds for restricted families of circuits as well. Fischer [Fis74] showed that for every poly-sized log-depth circuit, there is another equivalent poly-sized log-depth circuit which uses at most $\lceil \log(n+1) \rceil$ negations. A noteworthy point in Fischer's construction [Fis74] is that it is not planar; so it does not imply that evaluating planar circuits with $O(\log n)$ negations is P-hard. In contrast, Santha and Wilson [SW91] showed that there are functions requiring super-logarithmic number of negation gates in any poly-sized constant-depth circuit computing them. Our result can be viewed as a conditional topological analogue of this result, restricted to P-computable functions.

Let us try to evaluate a non-monotone planar circuit in parallel. The computation proceeds in stages. For any gate g where the subcircuit rooted at g has no negations, the value of g can be found in SAC^2, by Theorem 2.17. Assume that all such gates have been evaluated. Now let g be a gate such that in the sub-circuit rooted at g, a root-to-leaf path has at most one negation gate. Such gates can be evaluated by an SAC^2 circuit whose inputs include the original circuit input, the values of the gates already evaluated, and the negations of these values. Generalising this, we define *negation-height*, akin to the notion of cycle-height from the proof of Lemma 3.2.

Definition 3.4 (Negation Height). *The negation-height of an input gate (variable or constant) is 0, by convention. The negation-height of gate g is the smallest non-negative integer*

h such that every path from a leaf to g has at most h negation gates.

At stage k, we evaluate all gates at negation-height k. The inputs to the stage-k circuit are the circuit inputs, and the values as well as negated values of all gates at negation-height $j < k$. Each stage k has an SAC^2 circuit, obtained by putting together the SAC^2 circuits for each gate at negation-height $j < k$. Thus if gate g has negation-height k, then g can be evaluated by a polynomial-sized semi-unbounded circuit of depth $O(k \log^2 n)$.

Of course, this requires negation-height to be explicitly available. By placing a weight of 1 on edges out of a negation edge, and a weight of 0 on other edges, we see that negation-height of g is exactly the maximum weight of a g-to-leaf path. Since the circuit is a DAG, this is in NL, in SAC^2. So computing the negation-height is not a real bottleneck.

We thus have the following result:

Lemma 3.5. *A planar circuit in which the output gate is at negation-height k can be evaluated by a polynomial size semi-unbounded circuit of depth $O(k \log^2 n)$. Thus planar circuits with polylog negation-height can be evaluated in* NC . □

It is not necessary that the entire circuit be planar. Since the evaluation proceeds in stages, it is sufficient if for each h, the subgraph of all gates with negation-height h is planar. (It is easy to construct such circuits that are non-planar.)

Lemma 3.6. *A circuit C where*

1. *the output gate has negation-height k, and*
2. *for each $0 \leq h \leq k$, the subcircuit consisting of gates at negation-height exactly h is planar,*

can be evaluated by a polynomial size semi-unbounded circuit of depth $O(k \log^2 n)$. □

This result can be combined with the results of Sections 3.1 and 3.2. If the (output gate of the) circuit has negation-height $k \in O(\log^i n)$, and if for each $0 \leq h \leq k$, the subgraph of gates with negation-height exactly h is toroidal or multi-cylindrical, then the whole circuit can be evaluated in NC , *provided* the appropriate embedding for each subgraph is given. (Such embeddings are not explicitly required in proving Lemma 3.6, since planar embeddings can be constructed in L.)

3.4 Circuits with Limited Crossing Number

In this section we use the results in the previous sections and show that monotone circuits with polylogarithmic crossing number can be evaluated in NC.

We first define the notion of crossing number that we will be studying in this section.

Definition 3.7 (Crossing Number)**.** *The crossing number of a drawing of a graph is the total number of crossings of edges. The crossing number of G, $cr(G)$, is the smallest crossing number of any drawing of G.*

Notice that genus of graphs with crossing number k is at most k. However, we do not know if this inclusion is strict.

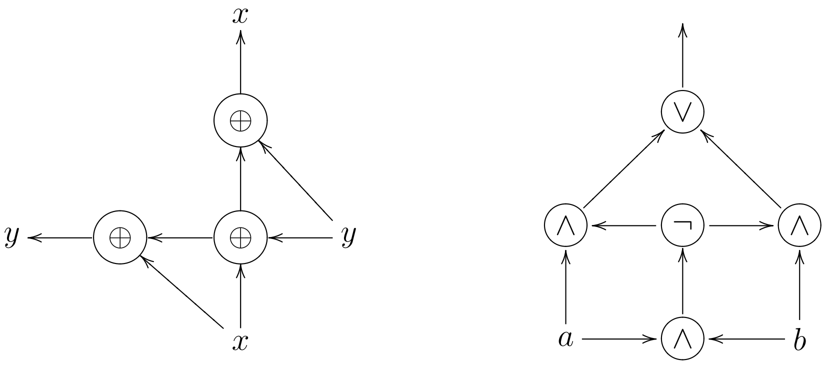
We consider monotone circuits which are not planar, but whose crossing number is limited in some way. Again, without such a limitation, there is no hope of evaluating the circuit inside NC unless P = NC, since monotone CVP is known to be P-complete [Gol77]. However, in this connection, it is worth noting that given any graph G computing the crossing number of G exactly is a hard problem [GJ92]. In addition, even with the promise that the crossing number of a graph G is k, obtaining the drawing of the graph which realises this crossing number is quite hard [Bie90]. But one may ask, in general, how many crossings are there in the graphs of the circuit corresponding to the P-hard instances?

The crossings associated with a combinatorial embedding can also be represented along with it. For each directed edge $e = (u, v)$ in the graph we can manintain an ordered list of edges which cross e, in the order in which they cross e while traversing from u to v.

Similar to the notions of cycle height and negation height in the previous section, we define the following:

Definition 3.8 (Crossing Height)**.** *The crossing number of a path of a graph in a given drawing is the total number of crossings in the embedding of the path. The crossing height of a vertex v in a drawing of an directed acyclic graph G, $cr(G)$, is the maximum crossing number of any path starting from v to a leaf in the drawing. Crossing height of a circuit with respect to a given drawing of the underlying directed acyclic graph G is the crossing height of the root vertex in the drawing.*

Notice that the crossing height of the root in the directed acyclic graph can be much less than the overall crossing number of the graph.



(a) Gadget with $\oplus$ gate (b) Implementing the $\oplus$ gate

Figure 3.4: Gadget which replaces a crossing with negations

Figure 3.4 shows a standard gadget for replacing a crossing in a non-planar drawing with two negations. Using this, we can easily derive the following lemma.

Lemma 3.9. *Let C be a circuit and let a drawing of C with respect to which C has crossing height of at most k be given. Obtain circuit C' by replacing each of the crossing in the drawing with the gadget in figure 3.4. Then, the negation height of C' is at most $2k$.*

Proof. We prove it by induction on k. Clearly if $k = 0$, the useful part of the circuit (which is reachable from the root) circuit is planar and nothing is to be done. Suppose it is true for $k' < k$, and consider the root gate of C'. (If there is more than one output, choose all the nodes which have the maximal crossing height, and apply the induction step to each of them). We can assume that the circuit is layered. Let r be the layer in which the first crossing occurs for the optimum layered embedding of the graph, and v be the corresponding vertex after this crossing. Thus crossing height of v is at most $k - 1$, and we can apply the induction hypothesis. Now replacing this particular crossing by the gadget of figure 3.4, will increase the negation height of root by at most 2, and hence the lemma follows. □

Noticing that C' is planar we can use Lemma 3.5 to get the following theorem.

Theorem 3.10. *A monotone circuit where the crossing height of the root is at most k can be evaluated by a semi-unbounded circuit of polynomial size and depth $2k \log^2 n$. When k is a constant this is in* SAC^2. *When k is* polylog *the problem has an* NC *upper bound.*

For applying the idea of Lemma 3.5, it is not necessary that the entire circuit be planar. Since the evaluation proceeds in stages, it is sufficient if for each h, the subgraph of all gates with negation-height h is planar. Using Lemma 3.6 we have the following:

Theorem 3.11. *A monotone circuit in which the crossing height of the root is at most k and for each $0 \leq h \leq k$, the subcircuit consisting of gates at crossing height exactly h is planar, can be evaluated by a semi-unbounded circuit of polynomial size and depth $\frac{k}{2}\log^2 n$, given an embedding which achieves this crossing height and planarity.*

We can extend this even further, to put simultaneous restrictions on both crossing number and the number of negations. This presumably is the largest circuit class that can be considered in this context.

Suppose that the root gate of the given circuit has negation height bounded by k_n and crossing height bounded by k_c. Then first apply the above construction to replace the crossing edges with corresponding gates there by incurring an additional negation height of two per crossing. This gives a planar circuit where the negation height of the root gate is at most $k_n + 2k_c$.

Theorem 3.12. *Let C be a circuit where the crossing height, and negation height of the root gate are at most most k_c and k_n respectively. Given an embedding which achieves this crossing number, C can be evaluated by a semi-unbounded circuit of depth $2(k_n + 2k_c)\log^2 n$. When k is a constant this is in SAC^2. When k is* polylog *the problem has an* NC *upper bound.*

Chapter 4

On the Thickness of Branching Programs

Since their introduction, the branching program model has been an object of much attention. A folklore result shows that polynomial-size branching programs decide exactly the languages in NL while counting paths in such branching programs characterises $\#\mathsf{L}$ and GapL. The surprising result of Barrington [Bar89] establishes that all of NC^1 can be captured by bounded-width branching programs (in fact, width 5 suffices). Subsequently, Caussinius et al [CMTV98] extended these results to the arithmetic setting and showed that width-6 branching programs capture GapNC^1.

Recently, there has been some work on topological restrictions of the underlying graphs in the context of circuits. Hansen [Han04] proved that constant width planar circuits capture exactly ACC^0. More recently, Allender et.al [ADR05a] extended this result, characterising ACC^0 using constant width poly-log genus circuits.

It is natural to ask similar questions in the case of branching programs too. In this direction, Barrington et.al [BLMS97] showed that constant width upward planar branching programs capture exactly AC^0. Recently, Hansen [Han04] proved that circuits which can be embedded on a cylinder can be computed in ACC^0.

We explore this thread further. In particular, we concentrate on another generalisation of the planarity criterion, namely *thickness* of the circuit. This has been already considered in [ADR05a] adopting a non-standard definition of thickness. We clean up the literature in this direction a bit, and compare with the standard definitions of thickness of circuits. Along the similar lines, Roy [Roy06] proves a thickness characterisation for L. We tighten these results and align them with the standard notions of thickness. In section 2, we introduce the definitions and in section 3, we prove the results on thickness of circuits and the connections to NC^1. The results in this chapter appear in [2].

4.1 Preliminaries

The *thickness* of a graph G, denoted $\theta(G)$, is the minimum number of planar subgraphs into which the edges of G can be partitioned. Equivalently, it is the minimum number of layers in a planar drawing of G, such that each edge belongs to a single layer, no two edges in the same layer cross, and edges are allowed to be drawn as arbitrary curves [Kai73].

The *geometric thickness* of a graph G, denoted $\bar{\theta}(G)$, is the minimum number of layers in a planar drawing of G, such that each edge belongs to a single layer, no two edges in the same layer cross, and edges must be drawn as straight line segments. Similar notions were also studied by Kainen [Kai73] and [DEH00].

Bernhart et.al. [BK79] define a related to geometric thickness as the *book thickness* denoted by $\mathsf{bt(G)}$. In this variation, the additional constraint is that the vertices of G must be placed in convex position. They also prove (see Lemma 2.1 of [BK79]) the equivalence of this definition of book thickness with the following more commonly used one, which we will also adopt for the purposes of the results in this chapter.

A book with k pages, a *k-book*, is a line L (called the spine) along with k non-intersecting half planes having L as their common boundary. For a graph G, $\mathsf{bt(G)}$ can be defined as the minimum number of half planes needed to embed the graph such that all the vertices of G can be placed in the line L and each edge is embedded on a single layer, no two edges in the same layer cross.

There are further variants of thickness considered in the literature; for instance, Wood [Woo01] considers layouts in which each edge is drawn with at most one bend, at which it may change layers. For more results on thickness, see the survey of Mutzel et al [MOS98].

Clearly, from these definitions, $\theta(G) \leq \bar{\theta}(G) \leq \mathsf{bt(G)}$, and these inequalities have been shown to be strict [DEH00]. In addition, Eppstein [Epp01] shows that the gap can be made arbitrarily large.

We will also need some complexity theoretic notions. A program over a semigroup $\langle \mathbb{S}, \circ \rangle$ of length ℓ is a sequence of instructions of the form $I_k = (x_i, a, b)$ where $a, b \in \mathbb{S}$, x_i is the i^{th} input and $1 \leq k \leq \ell$, where each instruction is interpreted as

$$\text{if } x_i = 1 \text{ then } a \text{ else } b$$

On an input instance, each of the instructions I_i will yield an element $z_i \in \mathbb{S}$ and

$$x \in L \iff \prod_{i=1}^{\ell} z_i = id$$

Permutation branching programs are layered graphs $G = (V, E)$ with $V_1, V_2 \ldots V_\ell$ as the vertices in each layer with $|V_i| = |V_{i+1}|$ for every i. Each edge is labelled with an input bit which decides whether it is active or not for the particular input. In addition, the edge set, that are active for each input, should be a permutation from V_i to V_{i+1} for each i.

To prove the claims about the classes NC^1 and L we will use the following complete languages for these classes.

Lemma 4.1 ([Bar89]). *Permutation branching programs of bounded width and polynomial size compute exactly the languages in* NC^1.

In other words, given a sequence of elements from a fixed non-solvable group, the problem of testing whether the product of the elements gives the identity or not, is complete for the class NC^1.

Cook and McKenzie [CM87] (see also [IL89]) proved that computing the iterated product of permutation matrices is complete for L. The permutation corresponding to each layer in the permutation branching program can be represented by permutation matrices by taking their bipartite adjacency matrix. If the permutation branching program is of polynomial size, these matrices will be of polynomial order. The operation of taking product of permutations can be represented as the product of permutation matrices. Thus we have the following lemma.

Lemma 4.2 ([CM87]). *Permutation branching programs of polynomial size compute exactly the languages in* L.

We will also use the following basic facts about permutations. Any element in S_n can be written as a product of transpositions. From this, the following fact is obvious. Let τ_u denote the transposition that exchanges elements in positions u and $u+1$; let σ denote the permutation $(12\ldots n)$ i.e a cyclic shift.

Fact 4.3. *Any permutation in S_n can be generated by the two permutations $\sigma = (1, 2 \ldots n)$ and $\tau_1 = (1, 2)$, and the chain of product is of length $O(n^2)$.*

The following observation is immediate. Consider the permutation represented as a bipartite graph. Even if we disallow edges being embedded outside the set of vertices, we have thickness 2 for these two special permutations. To impose this restriction, we talk about the layered thickness of the permutations.

Fact 4.4. *The bipartite graphs corresponding to the permutations σ and τ are of layered thickness 2, layered geometric thickness 2.*

The following figure illustrates this fact.

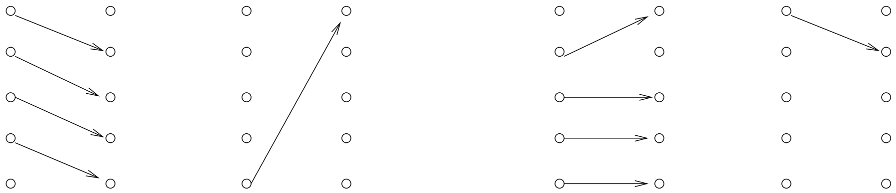


Figure 4.1: The first two are the two layers for σ and the last two are the two layers for τ.

4.2 Thickness of Branching Programs

The notion of thickness of graphs can be naturally extended to circuits too where we talk about the thickness of the underlying directed acyclic graph.

4.2.1 Thickness characterisation of NC^1 and L

The following proposition directly follows from Lemma 4.2 and facts 4.3 and 4.4.

Proposition 4.5. *1. Permutation branching programs of polynomial size and thickness 2 compute exactly the languages in* L.

2. Permutation branching programs of polynomial size and geometric thickness 2 compute exactly the languages in L.

Proof. By Lemma 2, it is sufficient to reduce the the problem of testing whether permutation branching programs accepts or not, to the corresponding thickness 2 BPs. The program consists of triplets of the form $\langle x_i, \theta, \delta \rangle$, which stands for the statement:

$$\text{If } x_i \text{ then } \theta \text{ else } \delta$$

where θ and δ are permutations in S_n.

Replace every such instruction by by the pair of instructions, (x_i, θ, id),$(\neg x_i, \delta, id)$.

Every permutation θ can be written as a product of transpositions $\theta_1, \ldots \theta_k$ where each θ_i is a permutation of the form τ_u for some u. So replace every statement of the form (x_i, θ, id) by the set of instructions $(x_i, \theta_1, id)(x_i, \theta_2, id) \ldots (x_i, \theta_k, id)$.

As in the case of facts 4.3 and 4.4. to write each $\tau_u (u > 1)$ as $\sigma^{n+1-u}\tau_1\sigma^{u-1}$. Replace every instruction of the form (l_i, τ_u, id) with $u > 1$ by the set of statements: $(l_i, \sigma, id)^{n+1-u}$, (l_i, τ_1, id), $(l_i, \tau_1, id)^{u-1}$. The following figure illustrates that the identity permutation can also be embedded on each layer of the branching program without increasing the thickness.

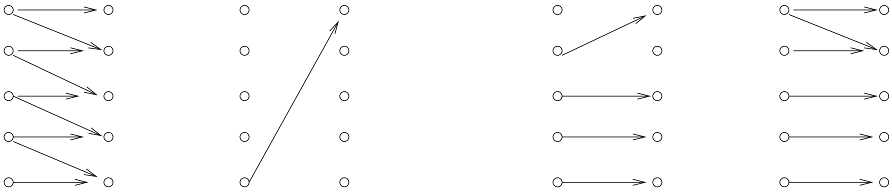


Figure 4.2: Identity permutation can be embedded in layer 1 of σ and layer 2 of τ without increasing their thickness.

To see the claim about geometric thickness, observe that as bipartite graphs, the permutations of transposition and cyclic shift are of geometric thickness 2 (see fact 4.3, and then using fact 4.3. □

Barrington characterised NC^1 in terms of bounded width branching programs over S_5. Notice that branching programs of width w can be of thickness w in general. However, using an argument similar to the above for programs over S_k where $k \geq 5$ is a constant, gives the following:

Proposition 4.6. *1. Permutation branching programs of bounded width, polynomial size and thickness 2 computes exactly the languages in* NC^1.

2. Permutation branching programs of bounded width, polynomial size and geometric thickness 2 computes exactly the languages in NC^1.

4.2.2 Page Characterisation of NC^1 and L

In this subsection we describe a characterisation of NC^1 recently proved by Allender et. al.[ADR05a]. They define the notion of *pages* as half planes joined in a common spine. A constant width circuit is of k pages if the vertices can be embedded on each of the half-planes(which are called pages) with the restriction that any subgraph embedded on each half plane is *upward planar*.

Theorem 4.7 ([ADR05a]). *Polynomial sized circuits of 3 pages with bounded width on each page recognise exactly the languages in* NC^1

However the terminology of *pages* used above is nonstandard. Notice the embedding on each of the edge can be made straight line upward planar embedding. Thus it is easy to see that graphs with k-page embedding have *geometric thickness* $k-1$ or less. Thus, in terms of the standard terminologies, the above result also implies proposition 4.6 (b).

Along, [Roy06] proves the following bound on number of pages needed to capture L.

Theorem 4.8 ([Roy06]). $\mathsf{L} = 4\text{-}\mathsf{PAGES}$.

However, this translates to a weaker bound in terms of geometric thickness compared to proposition 4.5.

Since book thickness can be strictly larger than geometric thickness, it is natural to look for book-thickness bounds for NC^1 and L. The proof of theorem 4.7 in [ADR05a] does not already give a book-thickness of 3, because for embedding each of the instructions for each stack, we would require to split up the edges into different layers for each of the pages.

4.2.3 Book-thickness characterisation of NC^1 and L

In this subsection we tighten proposition 4.5 further, by arguing that book thickness of 3 suffices. We will prove it in the most general form.

Theorem 4.9. *Every language in* L *can be accepted by a polynomial size branching program of book-thickness 3. In addition, the embedding is upward planar on each of the pages of the book.*

Proof. Again we start with the complete problem for L given by Lemma 4.2. By the proof ideas used in proposition 4.5, the permutation branching program can be written in such a way that each instruction is of the form (ℓ_i, σ, id) or (ℓ_i, τ, id) where ℓ_i is an input literal. Now insert a dummy instruction of the form (ℓ_i, id, id) between every two instructions of the program.

Call this resulting branching program P. This is directly embeddable in 3 pages: number the vertices (i, l) where i in $[n]$, l is the layer number. Place the vertices on the spine in layer-major order, with even-layer vertices ordered $1, 2, \ldots n$ and odd-layer vertices ordered $n, n-1, \ldots 2, 1$. Put the dummy layer edges on page 1. Put the real layer edges except edge $(2, 2i) - (1, 2i+1)$ or $(n, 2i) - (1, 2i+1)$ on page 2. Put the $(2, 2i) - (1, 2i+1)$ and $(n, 2i) - (1, 2i+1)$ edges where they exist on page 3. This is illustrated in figure 4.3 for the case of $n = 5$.

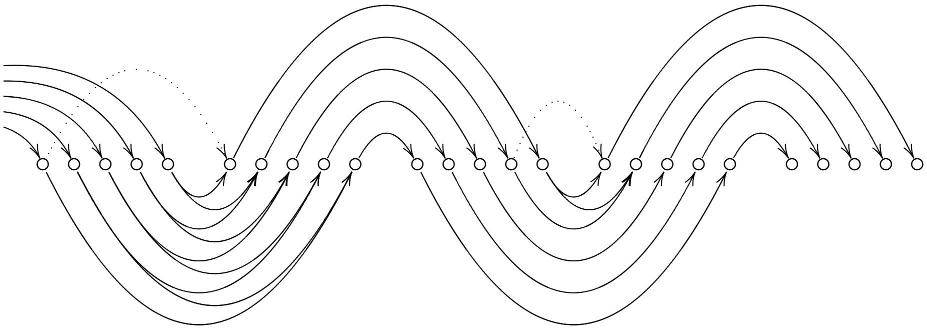


Figure 4.3: Arranging the instructions : $(\ell, \sigma, id), (\ell, id, id), (\ell, \tau, id), (\ell, id, id)$

□

Specialising the above arguments to the case of S_5, we have the following.

Theorem 4.10. *Every language in* NC^1 *can be accepted by a polynomial size branching program of width 5 and book-thickness 3. In addition, the embedding is upward planar on each of the pages of the book.*

Part II

Linear Algebraic Concepts Related to Circuit Complexity

Chapter 5

Circuit Complexity of Matrix Rank

Capturing the combinatorial notion of computation using algebraic problems is a promising area of research. Much of the work in this direction has been done with the view that the well-developed methods from algebraic structures could be used to obtain results about the power of the computational model. An important work [MS07] in this direction is the attempt to separate regarding the complexity of determinant and permanent of a matrix. In order to facilitate such approaches to major questions in space bounded boolean complexity theory, it will be useful to have algebraic problems capturing complexity classes. In this chapter we attempt this with the rank computation problem.

Many problems in Linear algebra have been shown to have efficient parallel algorithms. In particular, the complexity of computing the rank of a given matrix over a field $\mathbb{F}$ has been well studied. When the field under question is $\mathbb{Q}$, using a classic algorithm due to Csanky [Csa76], Ibarra et.al [IMR80] developed an efficient parallel algorithm (NC^2) to compute the rank of the matrix. Over arbitrary fields, a non-uniform NC^3 algorithm was developed by Chistov [Chi85]. Using a different approach, Mulmuley [Mul87] developed an NC^2 algorithm for the problem. Taking a more complexity theoretic perspective, building on Mulmuley's results, Allender et.al. [ABO96] showed that, for general matrices, computing the rank of a matrix over $\mathbb{Q}$ exactly can be done in $\mathsf{L}^{\mathsf{C_=L}}$. In addition, they showed that checking if the rank of a given matrix M is at most r is $\mathsf{C_=L}$-complete [ABO96], thus providing an exact complexity theoretic characterisation for the problem. It can be easily seen that their algorithm also works over the field $\mathbb{Z}_p$, and in this case the problem characterises $\mathsf{Mod}_p\mathsf{L}$ instead of $\mathsf{C_=L}$.

With the above motivation of characterising space bounded computation using algebraic problems, we study the problem of computing the rank under restricted settings.

The problem has been considered in the literature. In [BKR07], circuit complexity of computing the rank of restricted matrices is studied where the matrix entries are constrained by imposing constraints on the graph which it represents.

Here we take a different direction. The constraints that we impose on the matrix are more algebraic in nature. The main contribution in this chapter is the characterisation of deterministic log-space computation using rank computation problem for algebraically constrained matrices. In addition, we consider restrictions which are combinations of non-negativity, 0-1 entries, symmetry, diagonal dominance, tridiagonal and diagonal support, and we consider the complexities of three problems: computing the rank, computing the determinant and testing singularity. These, though intimately related, can have differing complexities, as Table 5.1 shows.

The results in this chapter appears in [4].

5.1 Basic Definitions

The notion of rank is a very basic concept in linear algebra. We start with the basic definition of rank of a matrix.

Definition 5.1. *Let M be an $n \times n$ matrix over a field $\mathbb{K}$. The following are equivalent and define the rank of the matrix over $\mathbb{K}$.*

1. *The size of the largest submatrix with a non-zero determinant.*
2. *The number of linearly independent rows/columns of a matrix.*
3. *The smallest r such that $M = AB$ where $A \in \mathbb{K}^{n \times r}, B \in \mathbb{K}^{r \times n}$.*
4. *The smallest k such that M is the sum of k rank 1 matrices, where rank 1 matrix is one in which there is a row(column) v such that each other row(column) is either a multiple of v or the zero vector.*

However, the above equivalence does not hold unless we are working over a field. We demonstrate this by an example. Consider a matrix: $\begin{pmatrix} 2 & 3 & 5 \\ 4 & 0 & 4 \\ 0 & 0 & 0 \end{pmatrix}$ over the ring $\mathbb{Z}_6$. By definition (4), the row rank of the matrix is 1, but the column rank of the matrix is 2.

The rank of a matrix over an integral domain I is same as that in the field of fractions of the $F(I)$. Indeed, any linear combination with coefficients from $F(I)$ can be translated to one in I by just clearing of the denominators. Thus, the rank of a matrix with rational entries is same over $\mathbb{Z}$ and $\mathbb{Q}$.

The rank of a matrix over a field $\mathbb{K}$ remains the same over a field $\mathbb{L}$ which contains $\mathbb{K}$. Indeed, when the determinant of a matrix over a field $\mathbb{K}$ is non-zero, it remains non-zero in any field $\mathbb{L}$ such that $\mathbb{K} \subseteq \mathbb{L}$. Thus for $M \in \mathbb{Z}^{n \times n}$, the rank is remains same over $\mathbb{Z}$, $\mathbb{R}$, $\mathbb{Q}$ and $\mathbb{C}$.

We consider the following computational problems.

$$\text{SINGULAR}(\mathbb{K}) = \{M \mid \text{ Over } \mathbb{K}, M \text{ is not full rank}\}$$

$$\text{RANK BOUND}(\mathbb{K}) = \{(M, r) \mid \text{ Over } \mathbb{K},\ \mathrm{rank}(M) < r\}$$

As discussed in the introduction, complexity theoretic characterisations for both these problems are known. (Note that for any type of matrices, and any complexity class $\mathcal{C}$, $\mathcal{C}$-hardness of SINGULAR implies $\mathcal{C}$-hardness of RANK BOUND.)

Proposition 5.2 ([ABO96]). *When $\mathbb{K} = \mathbb{Z}$ or $\mathbb{Q}$, SINGULAR$(\mathbb{K})$ and RANK BOUND$(\mathbb{K})$ and are complete for $\mathsf{C_{=}L}$.*

We now define some structural parameters of the matrix. In the following, let $\mathbb{K}$ be $\mathbb{Z}, \mathbb{Q}, \mathbb{R}$,or $\mathbb{C}$.

Definition 5.3. *(Diagonally Dominant) A matrix M over $\mathbb{K}$, is called* diagonally dominant *if:*

$$\forall i, |m_{i,i}| \geq \sum_{j \neq i} |m_{i,j}|.$$

If all the inequalities are strict, then M is said to be strictly diagonally dominant.

Here is an interesting property of strictly diagonally dominant matrices, follows from the standard Cholesky decomposition of matrices (See [MM64]). In the following we include a more direct proof.

Proposition 5.4. *([MM64]) If a matrix M is strictly diagonally dominant, then it is non-singular.*

Proof. Let M be strictly diagonally dominant matrix. For the sake of contradiction assume that M is singular. Thus there is a non-zero vector such that $Mx = 0$. Thus, for any i, $\sum_j m(i,j).x_j = 0$ Choose a k for which $|x_k| \geq |x_i|$ for all i. In particular for this k,

$|m(k,k)|.|x_k| = |-\sum_{j\neq k} m(k,j).x_j)| \leq |x_k| \sum_{j\neq k} |m(k,j)|$. This is a contradiction to strict diagonal dominance on the row k. □

Thus, if a matrix M is strictly diagonally dominant, the rank of the matrix is n and trivial to compute. As an example, consider the identity matrix I which is strictly diagonally dominant, and is of full rank. Now consider a matrix I_t where some of the non-zero entries on the diagonal are zeroed out. It is indeed true that the rank of the matrix is exactly n minus the number of such diagonal dominances which are not strict. An immediate question is about generalising this idea to arbitrary diagonally dominant matrices. We describe this briefly below.

Dahl [Dah99] established a strong connection between diagonal dominance on matrices and certain associated graphs defined as follows.

Definition 5.5. *For a non-negative symmetric diagonally-dominant matrix M, the* support graph *G_M is the undirected graph $G_M = (V, E_M)$ where $V = \{1, \dots n\}$, and $E_M = \{(i,j) \mid i \neq j,\ \ m_{i,j} > 0\} \cup \{(i,i) \mid m_{i,i} > \sum_{i\neq j} m_{i,j}\}$ (That is, self loops are added only at those vertices where the diagonal dominance is strict.)*

Lemma 5.6 ([Dah99]). *Let M be a non-negative symmetric diagonally dominant matrix of order n over $\mathbb{R}$. Then $\operatorname{rank}(M) = n - c$, where c is the number of bipartite components in the support graph G_M.*

Proof. This is reproduced from [Dah99] for completeness. Let $e_1, \dots, e_n$ be standard basis for the vector space $\mathbb{R}^n$. Let $\Delta^i = e_i e_i^T$, and $\Delta^{i,j} = (e_i + e_j)(e_i + e_j)^T$.

A set $X \subseteq \mathbb{R}$ is a cone if for $\alpha, \beta \geq 0,\ x, y \in X \implies \alpha x + \beta y \in X$. Extending this definition to $\mathbb{R}^{n\times n}$, we can prove that non-negative symmetric diagonally dominant matrices form a cone and any such matrix can be decomposed as:

$$A = \sum_{i=1}^{n} \left(a_{ii} - \sum_{j\neq i} a_{ij} \right) \Delta^i + \sum_{i<j} a_{i,j} \Delta^{i,j} \tag{5.1}$$

To analyse the rank of the matrix, it is natural to analyse how the matrix acts as a linear transformation on a vector $x = (x_1, \dots, x_n)$. Writing this down explicitly will give us:

$$x^T A x = \sum_{i=1}^{n} \left(a_{ii} - \sum_{j\neq i} a_{ij} \right) x_i^2 + \sum_{i<j} a_{i,j} (x_i + x_j)^2$$

When x is in the kernel, $Ax = 0$, and so,

$$x_i = \begin{cases} 0 & (i,i) \in E_M \\ -x_j & (i,j) \in E_M, i \neq j \end{cases}$$

Thus, for any component C in G_M, if there is odd cycle or a self loop for any vertex v, then $x_i = 0$ for all $i \in C$. For any bipartite component C in G_M, it is easy to see that if we choose a value for x_i for some $i \in C$, that will essentially determine the values of x_i for all $i \in C$. Thus, dimension of the null space of the matrix is exactly the number of bipartite components (c) and hence $rank(M) = n - c$. □

Remark 5.7. *When the matrix M is over $\mathbb{Q}$, using the fact that rank remains the same over $\mathbb{Q}$ and $\mathbb{R}$, the above Lemma also holds for $\mathbb{Q}$.*

Note that the presence of a self-loop means a component is non-bipartite. Hence the above lemma goes in accordance with the intuition that the rows in M for which diagonal dominance is not strict creates linear dependence in the subspace in corresponding connected component in G_M.

If all self-loops are present (that is, M is strictly diagonally dominant), then $c = 0$ and so M is non-singular. Thus it provides another proof of Proposition 5.4. (This latter result holds even if M is not symmetric or non-negative.)

Remark 5.8. *A natural question is about generalising the above lemma to matrices which are not symmetric. To this end, first we note that that the diagonally dominant matrices over $\mathbb{C}$ (even over $\mathbb{R}$) does not form a cone. But it is known that the set of diagonally dominant matrices with non-negative diagonal forms a cone([Dah99]). Define S_n to be set of matrices containing $e_ie_i^T$, $e_i(e_i + e_j)^T$ for every i. The techniques in Dahl [Dah99] also implies that the set of non-negative diagonally dominant matrices is exactly $cone(S)$. Using similar computations as in the proof of Lemma 5.6, it follows that the kernel of M will be exactly the $x = (x_1, \ldots, x_n)$ which are solutions of equations of the form*

$$a_{ii}x_i^2 = \sum_{i \neq j} a_{ij}x_ix_j$$

However, there does not seem to be a relation between x and the combinatorial structure of the graph associated with it.

Matrix type (over $\mathbb{Q}$)	RANK BOUND	SINGULAR	DETERMINANT
general (even 0-1)	C$_=$L-complete [ABO96]	C$_=$L-complete [ABO96]	GapL-complete [Dam91, Vin91] [Tod91, Val92]
symmetric non-negative	C$_=$L-complete [ABO96]	C$_=$L-complete [ABO96]	GapL-hard ($\leq_T^{\log}$ redn.) [Kul07]
symmetric non-negative diagonally dominant (d.d.)	L-complete	L-complete	?
symmetric diagonally dominant	L-hard even when $\det \in \{0,1\}$		?
symmetric d.d.	L-hard even when $\det \in \{0,1\}$		?
diagonal	TC^0-complete	AC^0	TC^0-complete
tridiagonal		C$_=$NC^1	GapNC^1
tridiagonal non-negative	non-negative permin planar # BWBP		

Table 5.1: RANK BOUND, SINGULAR, and DETERMINANT for special matrices

Complexity Theory Preliminaries: We refer to the appendix for the basic complexity theory definitions needed in this chapter. Computing the determinant over $\mathbb{Z}$ or $\mathbb{Q}$ is complete for GapL. In contrast, computing the permanent is complete for #P, the class of functions counting accepting paths of an NP machine. One of the classes figuring here that needs special explanation is planar #BWBP.

Branching programs as a computational model have been shown to be surprisingly powerful in the Boolean context; e.g. bounded-width branching programs (BWBP) capture NC^1, the class of languages polynomial size logarithmic depth circuits. However, in the arithmetic context, where we are interested in computing values rather than determining membership, they are not that well understood. It is still open ([All04, CMTV98]) whether the containment #BWBP $\subseteq$ #NC^1 is in fact an equality. It is known that width-2 layered planar #BWBP is at least as hard as NC^1 [AAB$^+$99]. Our results concerning tridiagonal and diagonal matrices give a simpler proof of a weaker result: width-2 layered planar #BWBP is at least as hard as TC^0.

5.2 Rank Computation for Diagonally Dominant Matrices

In this section we present the results on non-negative symmetric diagonally dominant matrices. To start our investigation about the combining the restrictions of various parameters, the following is easy to see: We include a proof for completeness.

Proposition 5.9. *The languages* RANK BOUND*($\mathbb{Z}$) and* SINGULAR*($\mathbb{Z}$) remain* $\mathsf{C_=L}$*-hard even if the instances are restricted to be symmetric 0-1 matrices.*

Proof. Let A' be the symmetric matrix $\begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix}$. Since $\mathrm{rank}(A') = 2(\mathrm{rank}(A))$, RANK BOUND($\mathbb{Z}$) remains $\mathsf{C_=L}$-hard when restricted to symmetric matrices. Further, DETERMINANT remains GapL hard even the matrices are restricted to be 0-1 (see for instance [Tod91]). Thus SINGULAR remains $\mathsf{C_=L}$-hard even when restricted to 0-1 matrices. Since M is in SINGULAR if and only if (M, n) is in RANK BOUND if and only if $(M', 2n)$ is in RANK BOUND, it follows that RANK BOUND($\mathbb{Z}$) remains $\mathsf{C_=L}$-hard for symmetric 0-1 matrices as well. □

This trick does not work for computing determinants, because $\det(A')$ will equal $\pm\det(A)^2$ and GapL is not known to be closed under taking square-roots. We do not know (any other way of showing) similar hardness for symmetric DETERMINANT. While it remains GapL hard for 0-1 matrices, it is not clear that there are GapL -hard symmetric instances. Recently, Kulkarni [Kul07] has observed that symmetric instances are GapL-hard under Turing reductions. The idea is to first use Chinese Remaindering: any determinant can be computed in L if its residues modulo polynomially many primes are available. Small primes (logarithmically many bits) suffice and can be obtained explicitly. Now to find the determinant modulo a small prime p, range over all $a \in \{0, 1, \ldots, p-1\}$ and test if it equals a modulo p. But this can be recast, using the GapL -completeness proofs of the determinant, as asking if a related determinant is 0 modulo p. Finally, using the idea in the proof of Proposition 5.9, we can ask the oracle for the determinant of a related symmetric matrix and test (in L) if it is 0 modulo p. We now consider an additional restriction where the matrix is *diagonally dominant*. We show:

Theorem 5.10. SINGULAR*($\mathbb{Z}$) and* RANK BOUND*($\mathbb{Z}$) restricted to non-negative diagonally dominant symmetric matrices are* L*-complete. The hardness is via uniform* AC^0*-computable many-one reductions.*

Proof. To show this, we exploit the connection between such matrices and their support graphs (see definition 5.5). For a matrix M, the *support graph* G_M is the undirected graph $G_M = (V, E_M)$ where $V = \{v_1, \ldots v_n\}$, and $E_M = \{(v_i, v_j) \mid i \neq j, \ \ m_{i,j} > 0\} \cup \{(v_i, v_i) \mid m_{i,i} > \sum_{i \neq j} m_{i,j}\}$. Now Lemma 5.6 essentially establishes that computing the rank of M is equivalent to counting the number of bipartite components in the support graph G_M.

Membership in L: Now, given a matrix M satisfying the stated conditions, it is straightforward to construct the support graph G_M. By [AG00, NTS95, Rei05], checking whether two vertices belong to the same component in an undirected graph, counting the number of components, and checking bipartiteness of a named component [JLL76] are all in L. Hence, by Lemma 5.6, $\mathrm{rank}(M)$ can be computed in L.
Hardness: The reduction is from undirected forest accessibility UFA, which is L-complete and remains L-hard even when the graph has exactly 2 components [CM87]. We state in the following lemma the special form that we need, and include the proof for completeness.

Lemma 5.11. *([CM87]) Given an undirected forest G, of bounded degree with exactly two components, and three special vertices s, t and q, with the guarantee that t and q are in different components, deciding which component s belongs to is L-hard.*

Proof. The reduction is from the machine model for L, and is essentially reproduced from [CM87]. We rephrase the proof here to highlight the fact that the normal form we need is indeed achievable.

To begin with, modify the machine description such that whenever the computation is on an infinite loop, the machine clears off the worktape and goes to an error state e. Thus there are only two poossible final states for the machine, one is the error configurations e, and the other is the accepting configuration t.

The set of configurations of a Turing machine with a fixed input w forms the vertices of such a graph G, and the (unique) accepting configuration is accessible from the initial configuration if and only if the Turing machine accepts the input w. G can be made acyclic by associating a time stamp with the configurations, and insisting that an edge always joins a configuration at time i to a configuration at time $i + 1$. If $p(n)$ is an upper bound on the computation time of the Turing machine with input w, then we let the node t in the graph be the accepting configuration with time stamp $p(n)$, and s will be the initial configuration with time stamp 0.

By definition, the number of possible (in/out)-neighbours of any node is bounded by a constant. In addition there are exactly two nodes of outdegree 0, and they correspond to the configurations e and t.

Viewing each edge in the resulting digraph as undirected yields an undirected forest such that s and t belong to the same tree if and only if a directed path existed from s to t in the original digraph. Note that the resulting undirected forest has precisely two components, and the three vertices satisfy the required properties of the reduction. □

We now construct G' as follows: Make two copies G_1 and G_2 of G. Add a new vertex u. Add edges $(s_1, s_2), (t_1, u), (t_2, u)$. Add self-loops at q_1 and q_2.

G' has at most three components (copies of the components containing t join up via u). The component(s) containing copies of q are necessarily non-bipartite.

If there is an $s \rightsquigarrow t$ path ρ in G, then in G' the two copies of the path, along with the edges $(s_1, s_2), (t_1, u), (u, t_2)$ create an odd cycle, so the new joined up component is also not bipartite. Hence G' has no bipartite components.

If there is no $s \rightsquigarrow t$ path in G, the component containing t_1 and t_2 will remain bipartite. Thus there is exactly one bipartite component now. To complete the proof, we need to produce a matrix M such that G' is its support graph. We construct M as follows:

$$\begin{array}{lll}\text{For each } i \neq j & m_{i,j} = & \begin{cases} 1 & \text{if } (i,j) \in E' \\ 0 & \text{otherwise} \end{cases} \\ \text{For each } i & m_{i,i} = & \begin{cases} 1 + \sum_{j \neq i} m_{i,j} & \text{if } (i,i) \in E' \\ \sum_{j \neq i} m_{i,j} & \text{otherwise} \end{cases}\end{array}$$

M can be constructed from G by a uniform TC^0 circuit. From Lemma 5.6, M is singular if and only if there is no $s \rightsquigarrow t$ path in G.

It is clear that M can be constructed from G', and hence from G, by a uniform TC^0 circuit.

Now we show that in fact it can be constructed in AC^0. First, observe that the forest that we start with (as the L-hard instance) has bounded degree. So we would like to rewrite the summation $\sum_{j \neq i} m_{i,j}$ as $\sum_{j \neq i; m_{i,j} \neq 0} m_{i,j}$. But how do we know *a priori* which entries are non-zero? For a node i, define L_i to be the list of nodes for which $m_{i,j}$ can possibly be non-zero. Since the log-space Turing machine alters only a small part of the configuration in one step, this list is of bounded length, with the bound l depending only on the machine's description and not on the input length. Let list(i, t) denote the t^{th} element in a lexicographical enumeration of L_i; on input i, t, list(i, t) can be determined in AC^0. Now the required summation is exactly $\sum_{j \in L_i} m_{i,j} = \sum_{t=1}^{l} m_{i,\text{list}(i,t)}$, and thus it can be computed by an AC^0 circuit. □

Corollary 5.12. *The language* RANK BOUND*($\mathbb{Z}$), restricted to symmetric non-negative diagonally dominant instances, is* L*-complete.*

However, the hardness of RANK BOUND($\mathbb{Z}$) is not just from the hardness of SINGULAR problem. An obvious way to obtain hardness at other values of rank (rather than $r = n$ in the case of SINGULAR) is to pad out the matrix with zero rows and/or columns. We present here a slight modification of the proof of Theorem 5.10 which establishes hardness of deciding whether the rank is $n-1$ or $n-2$.

Proof. The reduction is from undirected forest accessibility UFA, which is L-complete and remains L-hard even when the graph has exactly 2 components [CM87].

Let G, s, t be an instance of UFA, where G has two trees. We construct a new graph $G' = (V', E')$ as follows: take two disjoint copies of G. Add a new vertex u and connect it to both copies of t. Connect the two copies of s. Also, add self-loops at both copies of t.

If there is an $s \leadsto t$ path ρ in G, then G' has three components: the copies of the component containing s an t join up, while the copies of the other component remain disconnected (and hence bipartite). The two copies of the path, along with the edges $(s_1, s_2), (t_1, u), (u, t_2)$ create an odd cycle, so the new joined up component is not bipartite. Hence G' has exactly two bipartite components.

If there is no $s \leadsto t$ path in G, the component containing s_1 and s_2 will remain bipartite. The other component is not bipartite due to the self loops at $t_, t_2$. Thus there is exactly one bipartite component now.

To complete the proof, we need to produce a matrix M such that G' is its support graph. This can be done exactly in the same way as in the above proof. □

In Theorem 5.10, if we relax the condition of non-negativity, then the hardness of course continues to hold (but we do not know how to show membership in L). Via a somewhat different reduction, we show that for such matrices, L-hardness of SINGULAR holds even for matrices whose determinant is known to be in $\{0, 1\}$.

Theorem 5.13. SINGULAR*($\mathbb{Z}$) for symmetric diagonally dominant matrices is* L*-hard, even when restricted to instances with 0-or-1 determinant.*

Proof. As in the proof of Theorem 5.10, we begin with an instance (G, s, t) of UFA where G has exactly two components. Add edge (s, t) to obtain graph H. By the matrix-tree theorem, (see for e.g. Theorem II-12 in [Bol84]), if A is the Laplacian matrix of H (defined below), and B is obtained by deleting the topmost row and leftmost column of A, then $\det(B)$ equals the number of spanning trees of H.

$$\begin{aligned} a_{i,i} &= \text{ the degree of vertex } i \text{ in } H \\ a_{i,j} &= -1 \text{ if } i \neq j \text{ and } (i,j) \text{ is an edge in } H \\ a_{i,j} &= 0 \text{ if } i \neq j \text{ and } (i,j) \text{ is not an edge in } H \end{aligned}$$

Clearly, A is diagonally dominant (in fact, for each i, the constraint is an equality); also, since H is an undirected graph, A is symmetric.

Now the number of spanning trees in H is 1 if $s \not\rightsquigarrow_G t$ (H itself is a tree) and is 0 if $s \rightsquigarrow_G t$ (H still has two components). □

5.3 Determinant Computation of Special Matrices

Though rank for symmetric non-negative diagonally dominant matrices can be computed in L, we do not know how to compute the exact value of the determinant itself. In this section we address the question of determinant computation for restricted matrices. If a matrix is to have no trivial (all-zero) rows, and yet be diagonally dominant, then it cannot have any zeroes on the diagonal. How restrictive is this requirement? In general, it isn't too much so, as we show below. However, we do not know of a many-one reduction.

Lemma 5.14. *For every* GapL *function f and every input x, $f(x)$ can be expressed as* $\det(M) - 1$*, where M has no zeroes on the diagonal. Further, M can be obtained from x via projections.*

Proof. Consider Toda's proof [Tod91] for showing that DETERMINANT is GapL hard (see also [ABO96, MV97]). Given any GapL function f and input x, it constructs a directed graph G with self-loops at every vertex except a special vertex s. G also has the property that every non-trivial cycle (not a self-loop) in G passes through s. If A is the adjacency matrix of G, then the construction satisfies $f(x) = \det(A)$. Now consider the matrix B obtained by adding a self-loop at s. What additional terms does $\det(B)$ have that were absent in $\det(A)$? Such terms must correspond to cycle covers using the self-loop at s; i.e. cycle covers in $G \setminus \{s\}$. But $G \setminus \{s\}$ has no non-trivial cycles, so the only additional cycle cover is all self-loops, contributing a 1. Thus $\det(B) = 1 + \det(A)$, and B is the required matrix. □

Continuing further along restricting matrices, we consider the simplest form of the matrices considered in Theorem 5.10, namely non-negative diagonal matrices. Clearly, the rank is now the number of non-zero entries. Checking whether an entry is zero can be done by a single AND gate which looks at the negated literals in that entry. Since polylog thresholds are in AC^0 [RW91], it follows that not just singularity, but also instances (M, r) of RANK BOUND where r is within a polylog additive (subtractive) factor of 0 (or n, respectively) are in fact in AC^0. RANK BOUND($\mathbb{Z}$) itself, for such matrices, is in TC^0. Also, the determinant can be computed in TC^0 since it merely involves iterated multiplication.

On the other hand, an instance $x_1 \dots x_n$ of the TC^0-complete problem co -MAJORITY can be written as the instance $(D(x), n/2)$ of RANK BOUND($\mathbb{Z}$). ($D(x)$ is the matrix obtained by placing the vector x on the diagonal and placing zeroes elsewhere.) Similarly, an instance $a_1, \dots, a_n$ of iterated multiplication (n n-bit numbers) can be recast as such a determinant by placing the numbers on the diagonal. Thus

Theorem 5.15. *RANK BOUND($\mathbb{Z}$) and DETERMINANT, restricted to diagonal matrices, are TC^0-complete. The hardness does not require negative entries.*

This is another instance where RANK BOUND does not derive its hardness from the singularity threshold; it is in fact (provably) harder than SINGULAR. (The first instance is in the Note after Theorem 5.10; however, in that case, SINGULAR is also equally hard.)

The next restriction we consider is tridiagonal matrices: $m_{i,j} \neq 0 \implies |i-j| \leq 1$. We show that DETERMINANT and PERMANENT are in GapNC^1, by using bounded-width branching programs BWBP . In the Boolean context, BWBP equals NC^1. However, in the arithmetic context, they are not that well understood. It is still open ([All04, CMTV98]) whether the containment $\#\mathsf{BWBP} \subseteq \#\mathsf{NC}^1$ is in fact an equality (though it is known that $\mathsf{GapBWBP} = \mathsf{GapNC}^1$). Layered planar BWBP are the G-graphs referred to in [AAB$^+$99]. Counting paths in G-graphs may well be simpler than GapNC^1 due to planarity. However [AAB$^+$99] (see also [All04]) shows that even over width-2 G-graphs, it is hard for NC^1. We show that the permanent and determinant of tridiagonal matrices are essentially equivalent to counting in width-2 G-graphs. In what follows we have a weighted BWBP, where the weight of a path is the product of the weights of the edges on the path. The value of a weighted BWBP is the sum, over all s-t paths, of the weights of the paths.

Theorem 5.16. *Computing the permanent and determinant of a non-negative tridiagonal matrix over $\mathbb{Z}$ is equivalent to evaluating a layered planar weighted BWBP of width 2.*

Proof. Given a tridiagonal matrix A, let A_i be the top-left submatrix of A of order i, and let X_i and Y_i denote its permanent and determinant respectively. We have the following recurrences:

$$\begin{aligned} X_0 &= Y_0 = 1 & X_1 &= Y_1 = a_{1,1} \\ X_i &= a_{i,i}X_{i-1} + a_{i-1,i}a_{i,i-1}X_{i-2} & Y_i &= a_{i,i}Y_{i-1} - a_{i-1,i}a_{i,i-1}Y_{i-2} \end{aligned}$$

Figure 5.1 shows a weighted branching program for X_n that has width 2 and can be drawn in a layered planar fashion. The construction for the determinant Y_n is similar, using some negative weights. This completes the proof of one direction.

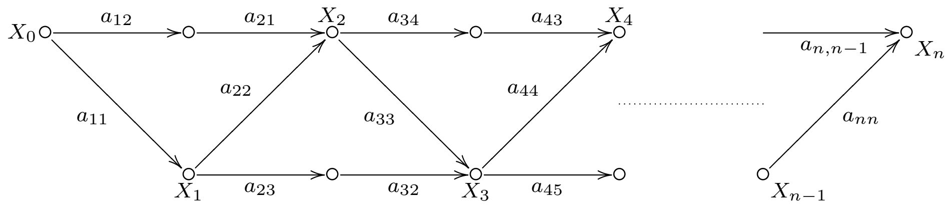


Figure 5.1: Width-2 branching program for tridiagonal permanent

We remark that in the construction for the permanent (X_n), when all entries are non-negative, this problem reduces to counting paths in unweighted planar branching programs of width 5. To see this, replace each weighted edge in Figure 1 with a width-three gadget having the appropriate number of paths in a standard way.

To see the other direction, notice that any layer of a width-2 planar BWBP should look like one of the following structures.

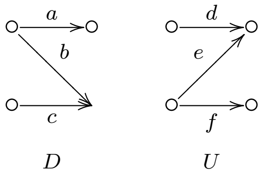


Figure 5.2: Components of Width-2 layered planar graphs

Now any width-2 graph G corresponding to the BP can be encoded as a sequence of D and U components as indicated in figure 5.2. First consider the case where the sequence consists of alternating D and U; that is consider sequences in $(DU)^*$. Each such sequence looks exactly like the graph in figure 5.1. By just reading off the weights on the corresponding edges in the graph, we can produce two matrices M_1 and M_2 such that permanent of M_1 and the determinant of M_2 (by putting in appropriate negations) equal to the value of the weighted BWBP.

Now it is sufficient to argue that the graph corresponding to any BWBP can be transformed to this form. If the string does not start with a with a D component, we will just put in a prefix D with $abc = 101$. Similarly, add a suffix U component with $def = 101$ if necessary. We need to handle the case when there are two consecutive components of the same type; UU or DD. Simply put in a D component with $abc = 101$ between two Us, and a U component with $def = 101$ between two Ds. Notice that the new width-2 graph when encoded will be an element of $(DU)^*$, and the weights of the paths are preserved in the transformation. The above reduction now gives the two matrices M_1 and M_2. In addition, observe that if the BWBP is unweighted, then the matrix M_1 that we produce has only 0,1 entries, and M_2 will have entries from $\{-1, 0, 1\}$. □

Corollary 5.17. *Computing the permanent and determinant of a tridiagonal matrix over $\mathbb{Z}$ is in* GapNC^1*, and is hard for* NC^1 *under* $\mathsf{AC}^0[5]$ *reductions.*

Chapter 6

Optimising Matrix rank

In this chapter we address the question of how *close* is a given matrix M to a matrix of given rank r. This is a natural optimisation question associated with matrix rank. The notion of *rank-robustness* of matrices has found many applications in circuit complexity, communication complexity, and learning complexity ([FKL$^+$01], [For02],[Raz89], [Lok95], [PP04], [LS06]). Historically, the problem was first studied by [EY36]. Besides the theoretical interests related to complexity theory, the related notion of rank robustness also finds practical applications; especially in Control Systems Design [Bol70, Bar75, BT00], Artificial Intelligence and Cognitive Sciences [BBF$^+$74].

This notion is also a close variant of the problem of *dimension reduction* which, besides its theoretical applications, has also received a lot of attention in applied areas in computer science such as machine learning [DM01], computer vision, and information retrieval[BDO95, SJ03]. In these contexts, the main aim is to obtain more compact representation of data with limited loss of information. A matrix of low rank, intuitively has a compact representation. Thus, approximating the given matrix (with an appropriate notion of distance to the original matrix) using another matrix of given rank, forms an important question in many areas.

6.1 Basic Definitions and Properties

In order to formulate our problem, we need to define the notion of *distance* between two matrices. The possible candidates for the notion of distance, comes from the matrix norms.

We first recall some basic definitions from matrix analysis. Let V be vector space V over a subfield $\mathbb{F}$ of the complex numbers. Let $|.|$ denotes the absolute value. A norm on V

is a function $p : V \to \mathbb{R}$; $x \mapsto p(x)$ with the following properties: (1) For all $a \in \mathbb{F}$ and all $u, v \in V$, $p(av) = |a|p(v)$, $p(u+v) \leq p(u) + p(v)$, (2) $\forall v \in V$, $p(v) = 0$ if and only if v is the zero vector. A norm of v is denoted by $||v||$, instead of $p(v)$. In particular, the p-norm of a vector $(x_1, \ldots, x_n) \in \mathbb{F}^n$ is denoted by, $||v||_p = (\sum_{i=1}^n x_i^p)^{\frac{1}{p}}$. $||v||_\infty = \max_i |x_i|$.

We start with a very general definition of matrix norm.

Definition 6.1. *Given two vector norms $||.||_\alpha$ and $||.||_\beta$ over vectors in $\mathbb{R}^n$, and a matrix $M \in \mathbb{R}^{m \times n}$, the subordinate matrix norm is defined to be:*

$$||M||_{\alpha,\beta} = \max_{||x||_\alpha = 1} ||Mx||_\beta$$

Based on choice of the norms α and β, we get different notions of distances. Indeed, for any choice of norms α and β, the above quantities forms norms of the spaces. For example, notice that $||M||_{1,\infty} = \max_{ij} |m_{ij}|$. In general when $\alpha = \beta$, we get the usual matrix norms.

$$||M||_1 = ||M||_{1,1} = \max_{||x||_1 = 1} ||Mx||_1 = \max_j \sum_i |m_{ij}|$$

is the *max* column vector norm.

$$||M||_2 = ||M||_{2,2} = \max_{||x||_2 = 1} ||Mx||_2 = \sqrt{\lambda_{\max}(M^T M))}$$

is the *spectral* norm.

$$||M||_\infty = ||M||_{\infty,\infty} = \max_{||x||_\infty = 1} ||Mx||_\infty = \max_i \sum_j |m_{ij}|$$

is the max row vector norm.

The *Frobenius* norm of the matrix is simply the ℓ_2 norm of the vector obtained by interpreting the matrix as an element in n^2-dimensional vector space. That is,

$$||M||_F = \sqrt{\sum_{ij} |m_{ij}|^2}.$$

From now on, unless otherwise stated, we assume $\alpha = \beta$ and denote the norm by $||M||_\alpha$. We will also assume that the matrix is a square matrix. We state a general version of our problem.

Problem 1. *Fix the vector norm $||.||_\alpha$ in $\mathbb{R}^n$. For any matrix $M \in \mathbb{R}^{n \times n}$ and an integer $r \leq n$,* *find a matrix $N \in \mathbb{R}^{n\times n}$ of rank at most r for which $||M - N||_\alpha$ is minimised.*

The problem has been well studied under several choices of the norm α. The most general exact answer known is in the case when $r = n - 1$.

Theorem 6.2 ([EY36]). *For any non-singular matrix $M \in \mathbb{R}^{n\times n}$, there exists a matrix N which is singular such that*

$$||M - N||_2 = \frac{1}{||M^{-1}||_2}.$$

In the cases when the norm α is Frobenius norm, the *singular values* (eigen values of MM^T) of the matrix M gives much information about the distance to the nearest singular matrix. Using this method, approximating a matrix with another one of low rank has also been considered in the literature under the name *low rank approximations* of matrices (see [FKV04] and references therein). A generalisation of the problem called subspace approximation has been considered in [DV07, Des07].

A natural variant of the problem is when we allow bounds on the component wise distance of the matrix. We say that $|M - N|_c \leq \delta$ when $\max_{ij} |m_{ij} - n_{ij}| \leq \delta$. This variant has also been well studied. Demmel [Dem92] provided some connections to the notion of *condition number* of the matrix. In a spirit similar to the Theorem 6.2, Poljack and Rohn [PR93] proved the following.

Theorem 6.3 ([PR93]). *For any non-singular matrix $M \in \mathbb{R}^{n\times n}$, there exists a matrix N which is singular such that*

$$|M - N|_c = \frac{1}{||M^{-1}||_{\infty,1}}.$$

Also, when different bounds are imposed on different entries (given by a matrix Δ), Poljak and Rohn [PR93] proved relationships to what are called signature matrices associated with the given matrix M. We skip more details and refer the reader to [Roh96].

A major component missing in the above considerations is the adaptability to the case of finite fields. The notion of distance with respect to norms is not defined over finite fields. In fact, the most natural notion of distance in the case of finite structures is that of *Hamming distance*. In a more algorithmic framework, the problem can also be viewed as the *edit distance* to the set of matrices of a given rank.

In the setting that we will be looking at, we are interested in constructing matrices such that there are no matrices at a *low* Hamming distance with rank less than a given value. We return to the computational question in this context later. We consider this in the following section under a more commonly known title.

6.2 Matrix Rigidity

A matrix is rigid if it is hard to lower its rank. More formally, the rigidity of a given matrix M with respect to a target rank r is the minimum number of entries that need to be changed in order to bring down the rank below r. Matrix rigidity was introduced by Valiant [Val77]; a similar notion was also studied independently by Grigoriev [Gri76]. In terms of standard notations:

Definition 6.4 (Matrix Rigidity)**.** *Over any field $\mathbb{K}$, the rigidity of a matrix $M \in \mathbb{K}^{n\times n}$ with respect to a target rank r is defined as:*

$$\mathrm{Rig}(\mathbb{K}, M, r) \overset{def}{=} \inf_{N} \left\{ \textit{support}(M-N) \mid N \in \mathbb{K}^{n\times n}, \mathrm{rank}(N) \leq r \right\}$$

where support(X) denotes the number of non-zero entries of the matrix X. When the field under question is clear, we denote the quantity by $R_M(r)$.

We now review some basic properties of this function. First of all notice that since it depends only on rank of the matrix, it is invariant under row and column permutations. The following bounds are well known.

Proposition 6.5 ([Val77])**.** *Over any field $\mathbb{F}$,*

$$\mathrm{rank}(M) - r \leq R_M(r) \leq (n-r)^2.$$

Proof. We use the following fact, which is folklore.

Fact 6.6. *Over any field $\mathbb{F}$, for any two matrices M and N of the same order,*

$$\mathrm{rank}(M) - \mathrm{rank}(N) \leq \mathrm{rank}(M+N) \leq \mathrm{rank}(M) + \mathrm{rank}(N).$$

The upper bound follows from sub-additivity property of matrix rank. To see the lower bound, write $C = M + N$. Thus $\mathrm{rank}(M) = \mathrm{rank}(C - N) \leq \mathrm{rank}(C) + \mathrm{rank}(N)$. Thus, $\mathrm{rank}(M+N) \geq \mathrm{rank}(M) - \mathrm{rank}(N)$.

It is immediate that if support$(M-N) = 1$ then $|\mathrm{rank}(M) - \mathrm{rank}(N)| \leq 1$. Thus, by changing an entry of a matrix we can change the rank by at most 1. This immediately shows that $R_M(r) \geq \mathrm{rank}(M) - r$.

Now we show the other direction. Without loss of generality we can assume that rank of M is at least $r+1$, otherwise the statement is vacuously true since $R_M(r) = 0$. Now, permute the rows and columns of the matrix such that the $r \times r$ submatrix at the top left corner is non-singular. Note that this is always possible since the rank of the matrix is at least $r + 1$. Now consider the bottom right $(n - r) \times (n - r)$ submatrix of M.

$$M = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$$

where $A \in \mathbb{F}^{r \times r}$, $B \in \mathbb{F}^{r \times (n-r)}$, $C \in \mathbb{F}^{(n-r) \times r}$, $D \in \mathbb{F}^{(n-r)\times(n-r)}$. Consider the $n \times r$ matrix obtained by joining A and C, which is rank exactly r. Thus we conclude that rows of C are expressible as the linear combinations of the rows of A. Let $c_1, c_2, \ldots c_{n-r}$ be the rows of the matrix C, $a_1, a_2, \ldots a_r$ that of A. For each i,

$$c_i = \sum_{j=1}^{r} \alpha_j a_j \tag{6.1}$$

Using the coefficients α_j, we change the rows of D to $d_1, \ldots, d_{n-r}$ to get matrix M', where:

$$d_i = \sum_{j=1}^{r} \alpha_j b_j$$

where $b_1, \ldots b_r$ are the rows of B. Notice that we changed only the entries of D in the above process. That is, we changed at most $(n-r)^2$ entries. By choice of the entries, every row of the matrix M' is expressible as a linear combination of the first r rows. Hence M' has rank at most r. □

The underlying field plays an important role in the above definition. If for a matrix M, we know that $k = \mathrm{Rig}(\mathbb{K}, M, r)$, over a field $\mathbb{K}$, a natural question is how does the value of rigidity in a subfield of $\mathbb{K}$, or in another field $\mathbb{F}$ which is an extension of $\mathbb{K}$. The following proposition follows from definitions

Proposition 6.7. *Let $\mathbb{K}$ be a field and $\mathbb{F}$ be an extension of $\mathbb{K}$, and let $M \in \mathbb{K}^{n \times n}$ then:*

$$\mathrm{Rig}(\mathbb{K}, M, r) \geq \mathrm{Rig}(\mathbb{F}, M, r).$$

An algebraic geometric formulation of the above problem is as follows. Consider a generic $n \times n$ matrix $X = (x_{ij})$. The condition of the rank of the matrix being at most r is equivalent to the condition that all the $(r+1) \times (r+1)$ minors of X are zero. This defines a set of $\binom{n}{r+1}$ polynomials whose common zeros are exactly the rank r matrices. Thus the set of rank r matrices forms an *affine variety* (for formal definitions see Appendix B). Indeed, matrix rigidity $R_M(r)$ is the hamming distance of M to this variety.

Bounded Rigidity: We will also consider the notion of bounded rigidity, which is related to the component-wise distance that was discussed in the previous section. Namely, changed matrix entries can differ from the original entries by at most a pre-specified amount θ. Let $\mathbb{K}$ be $\mathbb{R}$ or $\mathbb{Q}$ such that there is a notion of distance between two elements which is a norm.

Definition 6.8 (Bounded Rigidity). *The bounded rigidity of a matrix $M \in \mathbb{K}^{n\times n}$ with respect to a target rank r, and a $\theta \in \mathbb{K}$ is defined as:*

$$\mathrm{Rig}(\mathbb{K}, M, r, \theta) \overset{def}{=} \inf_N \ \left\{support(M-N) : N \in \mathbb{K}^{n\times n}, rank(N) \le r, \forall i,j : |m_{i,j} - n_{i,j}| \le \theta\right\}$$

where support(X) denotes the number of non-zero entries of the matrix X. When the field under question is clear from the context, we denote the quantity by $R_M(r,\theta)$.

Definition 6.9 (Norm Rigidity). *The norm bounded rigidity of a matrix $M \in \mathbb{K}^{n\times n}$ with respect to a target rank $r \le n$, and a $\theta \in \mathbb{K}$ is defined as:*

$$\Delta(\mathbb{K}, M, r) \overset{def}{=} \inf_N \left\{\sum_{i,j} |m_{i,j} - n_{i,j}|^2 : N \in \mathbb{K}^{n\times n}, rank(N) \le r\right\}$$

When the field under question is clear from the context, we denote the quantity by $\Delta_M(r)$.

The following lemma shows that the bounded rigidity functions can behave very differently from the standard rigidity function.

Lemma 6.10. *For any ϵ, and for any sufficiently large n such that $\frac{n}{\log n} > \epsilon + 1$, there is an $n \times n$ matrix M such that $R_M(n-1) = 1$, $\Delta_M(n-1) = \Theta(4^n)$, and the bounded rigidity $R_M(n-1, n^\epsilon)$ is undefined.*

Proof. Let M be an $n \times n$ diagonal matrix with $m_{i,i} = 2^n$ and $m_{i,j} = 0$ for $i \ne j$. Clearly, $R_M(n-1) = 1$; just zero out any diagonal entry. This involves a norm change of 4^n. Can M be made singular by a smaller norm-change, even allowing more entries to be changed? Recall the definition of strict diagonal dominance from Section 5.2. We invoke the Levy-Desplanques theorem (see for instance Theorem 2.1 in [MM64]) that says that the determinant of a strictly diagonally dominant matrix is non-zero. Now, a total norm-change less than 4^n will not destroy strict diagonally dominance, and the matrix will remain non-singular. Hence $\Delta_M(n-1) = 4^n$, and $R_M(n-1, n^\epsilon)$ is undefined. □

We will come back to this question again when we study computational question on rigidity in Chapter 8. As noted before the gap between the upper bound and lower bound for matrices in general is linear. In next two subsections we provide a quick survey of the applications of rank robustness in computational complexity theory.

6.3 Applications to Lower bounds

The notion of matrix rigidity found applications in complexity theory, in particular in the attempts to prove circuit lower bounds. In this section we briefly mention them.

To begin with, we recall the definition of arithmetic circuits. We will be interested in arithmetic circuits computing linear transformations, which are linear functions.

Definition 6.11 (Linear Circuits). *A linear circuit over a field K is a directed acyclic graph(DAG) with n source nodes (nodes of in-degree 0) and m sink nodes (nodes of out-degree 0) in which each edge is labelled by a non-zero element of $\mathbb{K}$. Each vertex of the graph denotes a gate g which takes in values from its in-coming edges labelled by $\alpha_1, \ldots, \alpha_k$ coming from gates $g_1, \ldots, g_k$, then g computes*

$$[g] := \sum_{i=1}^{k} \alpha_i [g_i]$$

where $[g_i] \in \mathbb{F}$ is the value computed at gate g_i. The size of a linear circuit is defined to be the number of edges in the graph. The depth of a circuit is the length of longest path from a source node to a sink node.

When the depth of the circuit is $\Omega(\log n)$, the fan-in of each gate can be assumed to be 2, otherwise, it is at least 2. Notice that each gate of a linear circuit is computing a linear function.

For a circuit C let $y_1, \ldots, y_m$ be the m sink nodes and $x_1, \ldots, x_n$ be the n source nodes. It is easy to see that since each gate computes a linear combination of its inputs, there exists a matrix A such that,

$$\begin{bmatrix} a_{11} & \ldots & a_{1n} \\ \vdots & \ldots & \vdots \\ a_{m1} & \ldots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}$$

where each a_{ij} is a linear combination of the α. More precisely, if we assume that the graph is layered into k layers, we can show that

$$a_{ij} = \sum_{p} \alpha_1 \ldots \alpha_k$$

where p ranges over all paths which takes source i to sink j.

The following result is well known, and establishes a very natural connection to rank of the matrix M and the combinatorics of the circuit that computes the linear transformation defined by M.

Lemma 6.12 (folklore,[Pud94]). *Let M be a matrix over a field $\mathbb{F}$. Let C any linear circuit computing the linear transformation corresponding to M, and let G be the corresponding directed acyclic graph. IF $\mathrm{rank}_{\mathbb{F}}(M) \leq r$ then G cannot have an $r-1$ vertex cut in the underlying directed acyclic graph. Conversely, if there are r vertex disjoint paths in G, then $rank_{\mathbb{F}}(M) \geq r$.*

Following the intuition behind this combinatorial connection, Valiant [Val77] proved the following. (The following version is stated in [PP04].)

Lemma 6.13 ([Val77],[PP04]). *Let r, δ, σ be positive integers with $k = \frac{\delta}{4\sigma} > 1$. Every linear transformation defined by an n-input m-output linear circuit over a field $\mathbb{F}$, with linear gates and with size at most $\frac{r \log \delta}{2 \log k}$ and depth at most δ can be written as*

$$A + BC$$

Here $B \in \mathbb{F}^{n \times r}$, $C \in \mathbb{F}^{r \times m}$ and the rows of A and B contains at most 2^σ non-zero entries.

Notice that if M is the matrix which defines the linear transformation computed by the linear circuit, then the above theorem immediately proves that there is a matrix A such that $M - A$ is a matrix of rank at most r. (Indeed, by the above theorem $M - A$ can be written as the product of an $n \times r$ matrix and an $r \times n$ matrix ; recall Definition 5.1). An interesting observation by [PP04] is that $M - A = BC$ where B is also sparse. Inspired by this observations, they studied a variant of rigidity. We refer the reader to [PP04].

We state the following as the corollary:

Theorem 6.14 ([Val77]). *If for some $\epsilon > 0$ there exists a $\delta > 0$ such that an $n \times n$ matrix M_n has rigidity $R_{M_n}(\epsilon n) \geq n^{1+\delta}$ over a field $\mathbb{F}$, then any linear circuit of depth $O(\log n)$ computing the transformation $x \to Mx$ has size $\Omega(n)$.*

In a similar note, [Pud94] also proves a connection between linear functions computed by bounded depth circuits and decomposition of the corresponding matrices. For $i \in [k]$, let C_i be a circuit which computes the linear transformation M_i. Then combining them in a linear fashion will give a circuit which computes the linear transformation $M = \prod_{i=1}^{k} M_i$. Using this connection, Pudlak [Pud94] proved the following implication for depth 2 circuits.

Theorem 6.15 ([Pud94]). *For every $\epsilon > 0$, there exists $\delta > 0$ such that for every $n \times n$ matrix over $\mathbb{F}$, $1 \leq m_1 \leq m_2 \leq n$,*

$$\begin{bmatrix} \forall r \in [m_1, m_2], \\ R_M(r) \geq \epsilon \frac{n^2}{r} \end{bmatrix} \implies \begin{bmatrix} \forall A, B : M = AB, \\ |A| + |B| \geq \delta n \log \frac{m_2}{m_1} \end{bmatrix}$$

where $|A|$ denote the number of non-zero entries in A. This also implies that every linear circuit of depth 2 which computes M must have size at least $\delta n \log \frac{m_2}{m_1}$.

For depth d, where d is a constant,

Theorem 6.16 ([Pud94]). *Let $\mathbb{F}$ be a field, and M be an $n \times m$ matrix. For every integer r, $1 \leq r \leq n$, for any constant d, any depth d circuit computing M must have size at least,*

$$\left(\frac{R_M(r)}{n}\right)^{\frac{1}{d}} r$$

We also state the applications of rigidity notion to communication complexity. Motivated by Yao's [Yao79] seminal work on the model of communication complexity, Babai et al. [BFS86] defined communication complexity analogs of of various complexity classes.

Let $\Sigma^{2*} = \{(x, y) \mid x, y \in \{0,1\}^*, |x| = |y|\}$ and consider any language $L \subseteq \Sigma^{2*}$. We can naturally represent its characteristic function on pairs of strings of length n as a $2^n \times 2^n$ boolean (or ± 1) matrix. On the other hand, given any $m \times m$ matrix ($m \geq 2^n$) we can associate a boolean function on $2n$ bits to it. (Note that this may not be unique as m may not be of the form 2^n for some n.) Now, [BFS86] defines the analog of PH as follows. Let m be a positive integer,

A language L is in Σ_k^{cc} if for some choice of $\ell_1(n), \ell_2(n), \ldots \ell_k(n)$, there exists boolean functions ϕ, ψ such that $(x, y) \in L_n$ if

$$\exists u_1 \forall u_2 \ldots Q_k u_k \left(\phi(x, u) \diamond \psi(x, u)\right)$$

where $u = u_1 \dots u_k$, $|u_i| = \ell_i(n)$, $\sum_i \ell_i(n) = \ell(n) \leq (\log n)^c$ for some constant $c \geq 0$, ϕ, ψ maps $n + \ell(n)$ bits to 1 bit, Q_k is $\forall, \exists$ and $\diamond$ is $\vee, \wedge$ when k is even and odd respectively. By allowing an bounded and unbounded (but not more than polylog) number of quantifiers we get PH^{cc} and PSPACE^{cc}, the communication complexity analogues of PH and PSPACE respectively. However, we do not know any explicit language outside Σ_2^{cc}. In the context, the following theorems show that lower bounds on rigidity of explicit 0-1 matrices would imply such separation results.

Theorem 6.17 ([Raz89]). *Let $\{A_n\}$ be an infinite family of 0-1 matrices over a finite field $\mathbb{F}$, and L_A be the associated language. For $r \geq (\log \log n)^{\omega(1)}$ if*

$$R_A(r) \geq \frac{n^2}{2^{(\log r)^{o(1)}}}$$

then $L_A \notin \mathsf{PH}^{cc}$.

Theorem 6.18 ([Lok95]). *Let $\{A_n\}$ be an infinite family of ± 1 matrices, and L_A be the associated language. For some constant c and for all constants $c_1, c_2 > 0$, if over the reals,*

$$R_A\left(2^{(\log \log n)^{c_1}}, 2^{(\log \log n)^{c_2}}\right) \geq \frac{n^2}{2^{(\log \log n)^c}}$$

then $L_A \notin \mathsf{PH}^{cc}$. In particular, if such a bound holds for the Hadamard matrix H for arbitrary c_1, c_2 and c, there is an explicit language in PSPACE^{cc} which is not in PH^{cc}.

The Challenge: As we described above, obtaining *strong* lower bounds for matrix rigidity and bounded rigidity for an *explicit* family of matrices, will yield lower bounds and separation results in various computational models. The notion of explicitness has appeared in many contexts in computer science, for example in the case of expander graphs, randomness extractors, linear codes.

As in many cases of problems regarding explicit constructions, the random matrix (under proper interpretations) achieves the desired rigidity bounds. The following was shown by Valiant in his seminal paper itself [Val77].

Theorem 6.19 ([Val77]). *Among all $n \times n$ matrices,* almost all *have rigidity $(n-r)^2$ if $\mathbb{F}$ is infinite, and $(n-r)^2/\log n$ when $\mathbb{F}$ is finite.*

Notice that over infinite fields, this is to be interpreted as the set of all rigid matrices is the complement of solution set of a finite system of algebraic equations. We will come back to a detailed proof of this, in the language of algebraic geometry in Section 7.3.2. To complete the picture, we may define the notion of explicitness a little more formally.

Definition 6.20 (Explicit). *An infinite family of matrices $\{A_n\}_{n \in I}, I \subseteq \mathbb{N}$ is* explicit *if there is a polynomial sized algebraic or Boolean circuit C which takes in as input n, i, j such that $1 \leq i, j \leq n$, tests if $n \in I$ and if yes, outputs $A_n(i, j)$.*

The above definition may be relaxed based on the application. For example, if we want to use Theorem 6.14 to obtain a lower bound for linear circuits over $\mathbb{C}$, we may allow the entries of the matrix to be arbitrary complex numbers. Thus they may not be polynomial time computable in the Turing machine model.

6.4 Previous Attempts on Lower Bounds

As we saw in the previous section, the main challenge about matrix rigidity is to obtain lower bounds for *explicit* matrices. Since strong lower bounds on rigidity will imply significant breakthroughs in complexity theory, many researchers [Raz89, SSS97, Fri93, Pud94, PR93, Lok95, Lok00, Lok06, LTV03] have explored the connections of this to various branches of mathematics. We refer the reader to [Cod00] and [Che05] for some comprehensive surveys on the topic. Several candidate matrices have been proposed, and many of them are conjectured to have quadratic rigidity. Table 6.1 shows what is known till date.

Out of the previous results mentioned in Table 6.1, the approaches in all attempts except 3,9-12, follows a common line of argument. Given a matrix M and a value r:

- First, show that *most* submatrices of size $r + 1$ have non-zero determinant.
- If you change too few entries, then there is a submatrix of size $r+1$ which will remain untouched, and hence the rank of the matrix will be at least $r + 1$, thus contradicting the rigidity requirement.

The first step varies from the choice of the candidate matrix. For example, for the case of Vandermonde matrices, [Pud94, Lok00] uses algebraic techniques, and for Hadamard matrices [KR97] uses techniques from functional analysis. But in a more general sense, the essence of this argument is captured by the proof of (5) due to [SSS97], where all the submatrices are non-singular.

No.	Family of Matrices (size n, $1 \leq i, j \leq n$)	Field $\mathbb{F}$	rank r	Best Bounds	References
1	$V = (a_i^{j-1})$ (Vandermonde)	$\mathbb{C}$		$\Omega(\frac{n^2}{r})$	[Pud94, Lok00]
2	V^{-1} (Inverse Vandermonde)	$\mathbb{C}$	any r	$\Omega(\frac{n^2}{r})$	[Raz89]
3	$V(X) = (x_i^{j-1})$ (Generic Vandermonde)	$\mathbb{C}$	$r \leq \sqrt{n}$	$\Omega(n^2)$	[Lok00]
4	$V(\zeta) = (\zeta^{(i-1)(j-1)})$ (Discrete Fourier Transform)	$\mathbb{C}$	$\log^2 n \leq r \leq \frac{n}{2}$	$\Omega(\frac{n^2}{r}\log(\frac{n}{r})$	[SSS97, Lok00]
5	$M\ :\ \lvert M^{-1}\rvert = n^2$ (Highly Non-singular)	$\mathbb{C}$	$\log^2 n \leq r \leq \frac{n}{2}$	$\Omega(\frac{n^2}{r}\log(\frac{n}{r})$	[SSS97]
6	$H : \lvert h_{ij}\rvert = 1 : HH^* = nI_n$ (Generalised Hadamard)	$\mathbb{C}$	$\log^2 n \leq r \leq \frac{n}{2}$	$\Omega(\frac{n^2}{r})$	[Alo94, Lok95] [KR97]
7	Parity Check Matrices of Linear Codes	$\lvert\mathbb{F}_q\rvert = q$	$\frac{2n}{\sqrt{q}-1} \leq r \leq \frac{n}{2}$	$\Omega(\frac{n^2}{r}\log\frac{n}{r})$	[Fri93, SSS97]
8	$C(X,Y) = (\frac{1}{x_i+y_j}) : \forall\, i,j :\ x_i \neq y_j$ (Cauchy Matrices)	$\lvert\mathbb{F}_n\rvert \geq 2n$	$\log^2 n \leq r \leq \frac{n}{2}$	$\Omega(\frac{n^2}{r}\log\frac{n}{r})$	[SSS97]
9	Triangular Shifters (Full 1s Lower Triangular)	Any $\mathbb{F}$	Any r	$\Omega(\frac{n^2}{r})$	[PV91]
10	Full 1s Extended Lower Triangular Matrices	Any $\mathbb{F}$	Any r	$\Omega(\frac{n^2}{r})$	[5]
11	$M(P) = (\sqrt{p_{ij}})$ p_{ij} are distinct primes	$\mathbb{C}$	Any r	$\Omega(n^2)$	[Lok06]
12	$M(P) = (e^{2\pi \mathrm{i}/p_{i,j}})$ p_{ij} are distinct primes of order n^{n^5}	$\mathbb{C}$	Any r	$(n-r)^2$	[6]

Table 6.1: Lower bounds on Matrix Rigidity : Current Snapshot

However, this strategy has the following drawback, as observed by Lokam [Lok00]. A theorem due to Lovasz [Lov75] implies the following:

Observation 1 ([Lok00])**.** *In any $n \times n$ matrix M, for any integer $0 < r < n$, there exists a set S of the entries of M such that, every $(r+1) \times (r+1)$ submatrix has at-least one entry in S, and*

$$|S| = O\left(\frac{n^2}{r}\log\frac{n}{r}\right)$$

Thus using step (2), we cannot hope to prove a lower bound for matrix rigidity better than $\Omega\left(\frac{n^2}{r}\log\frac{n}{r}\right)$. This provides a combinatorial barrier for the arguments which are based simply on the choice of entries.

The first technique which seems to implicitly surpass this barrier was developed by Lokam [Lok00] (based on the notion of algebraic dimension introduced in [SS91]) for the case of generic Vandermonde matrices (3). This was later extended [Lok06] to the case of matrices with entries based derived from prime numbers (rows 4,5 in Table 6.1). This approach penetrates the combinatorial barrier by essentially exploiting the algebraic structure of the entries of the matrix. However, a similar argument which works in the case of finite fields is not yet known.

We attempt to break this barrier by using geometric arguments about the space of rigid matrices. The study of geometry of matrix rigidity has been initiated in [LTV03]. We take a different approach and produce a family of matrices over $\mathbb{C}$ which achieves the maximum possible rigidity, $(n-r)^2$.

Intuitively, the idea is quite simple. Step (2) in earlier attempts essentially tried to make sure that the matrix is outside the rank r variety by proving the existence of an untouched non-singular submatrix of order $r+1$. Note that this could be thought of as trying not to satisfy one of the generators of ideal corresponding to the rank r variety.

We take a slightly different route to penetrate the combinatorial barrier. We attempt to directly argue that there are explicit matrices outside closure (in the Euclidean or even Zariski sense) the the set of rigid matrices. It suffices to show the existence of some *nice* polynomials with such properties which generate the ideal corresponding to the (Zariski) closure of the set of rigid matrices, and choose entries such that they are not satisfied under any choice of the changes. However, there are many technicalities to be settled in the above formalism. We set it up in the language of algebraic geometry, and use the machinery of elimination theory to exactly describe the variety which is the Zariski-closure of the set of matrices which are not maximally rigid. This formulation helps us to come up with a “low-degree” polynomial with properties and hence choose a matrix which does not satisfy it. These results are described in Chapter 7. However, we do not know how this approach compares with that of [Lok06].

6.5 An Almost Tight Bound for the Full 1s ELT Matrix

As we saw in the previous section, obtaining explicit bounds on the rigidity of special matrices is surprisingly elusive, and thus has received a lot of attention. The rareness of matching, or even close, lower and upper bounds, correlates well with the lack of upper bounds on the computational version of rigidity Chapter 8.

A rare case where a closed-form expression has been obtained for rigidity is full-1s triangular shifters (lower triangular matrices) [PV91, SK92]. An interesting point about their proof [PV91] is that it is completely combinatorial and does not follow the two step line of argument that we described in the previous section. In addition, it proves that the known rigidity bounds of many matrices can be achieved by a simple matrix: full 1s triangular shifters.

The rigidity is for the full-1s lower triangular matrices T_n: T_n is the matrix of order n with $j \leq i \Longrightarrow m_{i,j} = 1$, $j > i \Longrightarrow m_{i,j} = 0$. It is shown in [PV91] that over any field,

$$R_{T_n}(r) = \frac{(n - r + \Delta)(n + r - \Delta + 1)}{2(2r + 1)}$$

where $n = 2rk + r + k + \Delta$ for $k \geq 0$, $1 \leq \Delta \leq 2r + 1$.

In this section we consider an extension of the result of [PV91] to full-1s extended lower triangular (*elt*) matrices. In an elt matrix, the first diagonal above the main diagonal can be non-zero, but all other elements above the diagonal must be 0. (That is, $m_{i,j} \neq 0 \Longrightarrow j \leq i + 1$.) It is worthwhile noting that elt matrices can capture a lot of information: it is known that determinant/permanent computation of elt matrices is as hard as the general case, see [AAM03, Li92]. A full-1s elt matrix EL_m of order m is an elt matrix satisfying $j \leq i + 1 \Longrightarrow m_{i,j} = 1$. Even with this small extension beyond T_m, we cannot obtain a closed-form expression for rigidity; however, we show lower and upper bounds differing by an additive factor of roughly n/r. It may be worthwhile noting that even this simple family of matrices, also achieves the rigidity bounds known for many families of matrices.

Theorem 6.21. *Given n and r such that $r \leq n - 2$, define the following quantities: $k = \lfloor \frac{n-r-1}{2r+1} \rfloor$; $\delta = n - r - k(2r + 1)$; $\Gamma = \frac{(k+1)}{2}(n - r + \delta)$; $\ell = \lfloor \frac{n-r}{2r+1} \rfloor$. Now, over any field,*

1. *If $n \leq 3r$, then $R_{EL_n}(r) = n - r - 1$.*
2. *If $n \geq 3r + 1$, then $\Gamma \leq R_{EL_n}(r) \leq \Gamma + \ell - 1$.*

Our upper bound proof directly mimics that of [PV91]. Our lower bound proof mimics that of [PV91] to obtain one bound, and then further tightens it when $n = 3r + 1$. A combinatorial argument that can provide a similar tightening at all $n = r + k(2r + 1)$ would completely close the gap between the upper and lower bounds, but we do not see how to obtain this.

Upper Bound: Define $\tau = n - r - (2r + 1)\ell$. We will show that

$$R_M(r) \leq \frac{(\ell + 1)}{2}(n - r + \tau) + \ell - 1$$

This immediately yields the claimed upper bound when $n \leq 3r$. When $n \geq 3r+1$, consider two cases:

Case 1: $\ell = k$. Then $\tau = \delta$ and so $\Gamma = \frac{(k+1)}{2}(n-r+\delta) = \frac{(\ell+1)}{2}(n-r+\tau)$.

Case 2: $\ell = k+1$. Then $\tau = 0$, $\delta = 2r+1$, and $n = 2r\ell + r + \ell = \delta\ell + r$. So

$$\begin{array}{rcl} \Gamma & = & \frac{(k+1)}{2}(n-r+\delta) \\ & = & \frac{(\ell+1)}{2}(n-r+\delta) - \frac{1}{2}(n-r+\delta) \\ & = & \frac{(\ell+1)}{2}(n-r+\tau) + \frac{(\ell+1)}{2}(\delta) - \frac{1}{2}(\delta\ell+\delta) \\ & = & \frac{(\ell+1)}{2}(n-r+\tau) \end{array}$$

Thus in either case, the upper bound holds.

Now we establish the upper bound in terms of ℓ and τ.

We start with the matrix EL_n, of rank $n-1$. We identify r linearly independent rows $R_{j_1}, \ldots R_{j_r}$ which we will keep intact, so the rank of the resulting matrix is still at least r. We will change each of the other rows to one of these rows by changing some entries. But to minimise the number of entries changed, we adopt the following general strategy used in [PV91] for T_n. Let n_0 be the first set of rows which we will explicitly make zero. Similarly, n_{2i-1} is the number of rows just above R_{j_i} which are changed to R_{j_i} by changing the appropriate 0s to 1s, and n_{2i} is the number of rows below the row R_{j_i} which are changed to R_{j_i} by changing the appropriate 1s to 0s. Now the total number of changes is a function of these n_i's, as described below, and the natural idea for minimising the number of changes be to make the contribution of each n_i roughly equal. In particular, this evenly spaces out the rows to be preserved. In detail:

$$\begin{array}{rclcl} \text{\# of changes in } n_0\text{-block} & = & \sum_{t=1}^{n_0}(t+1) & = & \frac{n_0(n_0+3)}{2} \\ \text{\# of changes in } n_{2i-1}\text{-block} & = & \sum_{t=1}^{n_{2i-1}} t & = & \frac{n_{2i-1}(n_{2i-1}+1)}{2} \\ \text{\# of changes in } n_{2i}\text{-block} & = & \sum_{t=1}^{n_{2i}} t & = & \frac{n_{2i}(n_{2i}+1)}{2}) \\ \text{\# of changes in } n_{2r}\text{-block} & = & n_{2r} - 1 + \sum_{t=1}^{n_{2r}-1} t & = & \frac{(n_{2r}+2)(n_{2r}-1)}{2} \end{array}$$

and we want to minimise the total number of changes.

It can be seen that the optimal choice to achieve this would be to make all the n_i's equal, except n_0 which should be one less. This can happen when $\tau = 2r$; we set $n_0 = \ell$, $n_i = \ell + 1$ for $i \geq 1$. When $\tau < 2r$, some of the blocks other than n_0 will also have size ℓ rather than $\ell + 1$. We let the last τ blocks have size $\ell + 1$, and the first $(2r + 1 - \tau)$ be of size ℓ. Thus,

$$\begin{aligned}\text{Total number of changes} &= \tfrac{\ell(\ell+1)}{2}(2r+1) + \ell - 1 + (\ell+1)\tau \\ &= \tfrac{(\ell+1)}{2}\left[n - r + \tau\right] + \ell - 1\end{aligned}$$

Lower Bound: The lower bound when $n \leq 3r$ is easy to see: for decreasing the rank of any matrix, at least one entry has to be changed.

The lower bound when $n \geq 3r+1$ is a little more tricky. In [PV91], the corresponding lower bound for lower triangular matrices T_n is obtained by first showing that if $T_n + B_n$ has rank bounded by r, then some row of B_n has at least $k+1$ non-zero entries. Deleting this row and column yields $T_{n-1} + B_{n-1}$ also of rank bounded by r. Applying this argument repeatedly, the total number of changes is bounded by a certain sum, yielding the result. Our proof follows the same outline, and differs in essentially two places: (a) Deleting any row i and column $i+1$ of EL_n yields EL_{n-1}. (b) At $n = 3r+1$ a tighter bound is possible.

Given r, for each n we define

$$k_{n,r} = \left\lfloor \frac{n-r-1}{2r+1} \right\rfloor; \qquad \delta_{n,r} = n - r - k_{n,r}(2r+1)$$

Thus $k_{n,r}(2r+1) + r + 1 \leq n \leq k_{n,r}(2r+1) + 3r + 1$. The value of $k_{n,r}$ remains unchanged for $2r+1$ successive values of m, during which $\delta_{m,r}$ ranges over 1 to $2r+1$.

Notice that if $r+2 \leq n \leq 3r+1$, there is a row with at least 1 change. Now, for a general n, assuming that $EL_n + B_n$ has rank bounded by r, repeated applications of the following lemma show that B_n has reasonable row-wise density.

Lemma 6.22. *Let $r \leq n-2$, and let B_n be a matrix such that* $\mathrm{rank}(EL_n + B_n) \leq r$. *Let $k = k_{n,r}$, $\delta = \delta_{n,r}$. Then some row in B_n, other than the last row, has at least $(k+1)$ non-zeroes.*

Proof. This proof is similar to that in [PV91]. Assume to the contrary that every row of B_n (possibly other than row n) has fewer non-zeroes than required. Let $A_n = EL_n + B_n$. The idea is to choose a set S of $r+1$ rows which exclude row n, are linearly independent in EL_n, and are linearly dependent in A_n, and to then show that one of the rows from S in B_n has many non-zeroes. We choose S as follows

$$S = \{k, k+(2k+1), \ldots, k+r(2k+1)\}$$

Since $\mathrm{rank}(A_n) \leq r$, the rows indexed by S are linearly dependent in A_n; hence for some non-empty subset S' of S, we have non-zero α_j's satisfying

$$\sum_{j\in S'} \alpha_j a_j = 0 \qquad \text{and hence} \qquad \sum_{j\in S'} \alpha_j l_j = -\sum_{j\in S'} \alpha_j b_j$$

Here a_j, l_j, b_j refer to the jth row vectors of A_n, EL_n and B_n respectively. By our assumption, the vector on the right-hand-side RHS has at most $s'k$ non-zero entries ($s' = |S'|$). Exploiting the special structure of the matrix, we show that the left-hand-side LHS has more non-zero terms than the RHS and get a contradiction. Due to the structure of EL_n, the LHS is of the form $(c_1, c_1 \ldots c_1, c_2, c_2 \ldots c_2, \ldots c_{s'} \ldots c_{s'}, 0 \ldots 0)$. Each c_i section is of size at least $2k+1$, except the c_1 section, which has size at least $k+1$. Two consecutive sections cannot be zeros since $\alpha_j \neq 0$ for all j. And the last section necessarily has $c_{s'} \neq 0$.

Case 1: $s' = 2t+1$ for some t. Now consider the LHS. There are at least $t+1$ blocks of non-zeroes. At most one of these (the first) is of size $k+1$; all the rest have size $2k+1$. Hence the number of non-zero elements on the LHS is at least $(2k+1)t + k + 1 = (2t+1)k + t + 1 > s'k$.

Case 2: $s' = 2t$ with $t \neq 0$. There are at least t blocks of non-zeros. Furthermore, if the first block is a non-zero block, then in fact there must be $t+1$ non-zero blocks. Thus there are at least t blocks of non-zeros of size $2k+1$. Thus the number of non-zeroes on the LHS is at least $t(2k+1) > s'k$.

□

Lemma 6.23. $R_{EL_n}(r) \geq 2r+1$ *when* $n = 3r+1$.

Proof. Suppose not; assume that $2r$ changes suffice to bring the rank of $E = EL_{3r+1}$ to r or less. That is, there is a matrix B with at most $2r$ non-zero entries such that $A = B + E$ has rank r or less. Since there are $3r+1$ rows, at least $r+1$ of them are left unchanged. These must be linearly dependent to achieve rank(A) $\leq r$, so they must include rows $n-1$ and n of E (all other rows of E are linearly independent) and exactly $r-1$ other rows.

Let S be the set of preserved rows; $|S| = r+1$ and $\{n-1, n\} \subseteq S$. Let $S' = [n] \setminus S$; then $|S'| = 2r$. Each row of B in S' has at least one non-zero. But since there are only $2r$ non-zeroes overall, each row in S' has, in fact, exactly one non-zero.

For each $i \in S'$, row i is dependent on S and on $S \setminus \{n\}$. (With a single change per row, no row cannot be zeroed out.) Let $U = (S \setminus \{n\}) \cup \{i\}$. Then, as in the proof of

Lemma 6.22, there exists $U' \subseteq U$: $i \in U'$, and for each $u \in U'$, $\exists \alpha_u \neq 0$ such that

$$\sum_{u \in U'} \alpha_u e_u = - \sum_{u \in U'} \alpha_u b_u.$$

The RHS has a single non-zero in row i since rows of B from S are zero. The LHS is of the form $(c_1, c_1 \ldots c_1, c_2, c_2 \ldots c_2, \ldots c_{u'} \ldots c_{u'}, 0 \ldots 0)$ where $c_{u'} \neq 0$. To get just one non-zero on the LHS, $c_{u'}$ must be a block of size 1, and all other c_j's must be zero. Thus $\exists k : U' = \{k-1, k\}$, and $\alpha_k + \alpha_{k-1} = 0$. But, we know that α_i must be non-zero, since this is the row we are expressing as a combination of rows in S. Hence U' must be either $\{i-1, i\}$ or $\{i, i+1\}$. Thus, for each row $i \in S'$, either row $i-1$ or row $i+1$ is in S. So rows in S can be separated by at most 2 rows of S'. Since rows $n = 3r+1$ and $n-1 = 3r$ are in S, the 3rd last row of S is at least $3r-3$, the 4th last row of S is at least $3r-6$, and so on; the first row of S is at least row 3. But then row 1 does not have a neighbouring row in S, a contradiction. □

Using these lemmas we can establish the lower bound. When $n \geq 3r+2$, apply Lemma 6.22 repeatedly, eliminating one dense row each time, preserving the ELT structure, until n comes down to $3r+1$. Now Lemma 6.23 says that $2r+1$ more changes are necessary. Thus the total number of changes is at least $\delta(k+1) + (2r+1)k + (2r+1)(k-1) + \ldots + (2r+1)3 + (2r+1)2 + (2r+1) = \frac{(k+1)}{2}(n - r + \delta)$, giving the lower bound.

Chapter 7

Lower bounds for Matrix Rigidity

In this chapter, we describe the construction of an infinite family of complex matrices with the largest possible, i.e., $(n-r)^2$, rigidity. The entries of an $n \times n$ matrix in this family are distinct primitive roots of unity of orders roughly n^{4n^4}. These matrices, though not entirely explicit, do have a succinct algebraic description.

Our construction is based on elimination theory of polynomial ideals. In particular, we use results on the existence of polynomials in elimination ideals with effective degree upper bounds (effective Nullstellensatz). Using elementary algebraic geometry, we prove that the exact dimension of the affine variety of matrices of rigidity at most k is $n^2 - (n - r)^2 + k$. Finally, we use elimination theory to examine whether the rigidity function is semicontinuous.

The results in this chapter appears in [6].

7.1 Introduction

As we saw in the previous chapter, the study of lower bounds on rigidity of explicit matrices are motivated by their numerous applications in complexity theory. Over finite fields, the best known lower bound for explicit A was first proved by Friedman [Fri93] and is $\mathrm{Rig}(A, r) = \Omega(\frac{n^2}{r} \log \frac{n}{r})$ for parity check matrices of good error-correcting codes. Over infinite fields, the same lower bound was proved by Shokrollahi, Spielman, and Stemann [SSS97] for Cauchy matrices, Discrete Fourier Transform matrices of prime order, and other families. Note that this type of lower bound for high rank ($\mathrm{rank}(A) = \Omega(n)$) matrices reduces to the trivial $\mathrm{Rig}(A, r) = \Omega(n)$ when $r = \Omega(n)$. In [Lok06], lower bounds (over $\mathbb{C}$) of the form $\mathrm{Rig}(A, \epsilon n) = \Omega(n^2)$ were proved when $A = (\sqrt{p_{jk}})$ or when

$A = (\exp(2\pi \mathrm{i}/p_{jk}))$, where p_{jk} are the first n^2 primes. These matrices, however, are not explicit in the sense defined in Section 6.2 (Definition 6.20).

In this chapter, we construct an infinite family of complex matrices with the highest possible, i.e., $(n-r)^2$ rigidity. The entries of the $n \times n$ matrix in this family are primitive roots of unity of orders roughly n^{4n^4}. We show that the real parts of these matrices are also maximally rigid. Like the matrices in [Lok06], this family of matrices is not explicit in the sense of efficient computability described earlier. However, as mentioned above, one of the motivations for studying rigidity comes from algebraic complexity. In the world of algebraic complexity, any element of the ground field (in our case $\mathbb{C}$) is considered a primitive or atomic object. In this sense, the matrices we construct are explicitly described algebraic entities. To the best of our knowledge, this is the first construction giving an infinite family of non-generic/concrete matrices with maximum rigidity. It is still unsatisfactory, though, that the roots of unity in our matrices have orders doubly-exponential in n. Earlier constructions in [Lok06] use roots of unity of orders $O(n^2)$ but the bounds on rigidity proved there are weaker: $n(n-cr)$ for some constant $c > 2$.

Our approach to studying rigidity is based on elementary algebraic geometry and elimination theory. To set up the formalism of this approach, we begin by reproving Valiant's result that the set of matrices of rigidity less than $(n-r)^2$ form a Zariski closed set in $\mathbb{C}^{n \times n}$, i.e., such matrices are solutions of a finite system of polynomial equations (hence a generic matrix has rigidity at least $(n-r)^2$). In fact, we prove a more general statement: the set of matrices of rigidity at most k has dimension (as an affine variety) exactly $n^2 - (n-r)^2 + k$. This sheds light on the geometric structure of rigid matrices. Our transversality argument in this context is clearer and cleaner than an earlier attempt in this direction (in the projective setting) by [LTV03]. To look for specific matrices of high rigidity, we consider certain elimination ideals associated to matrices with rigidity at most k. A result in [BMMR02] using effective Nullstellensatz bounds [Bro87], [Kol88]) shows that an elimination ideal of a polynomial ideal must always contain a nonzero polynomial with an explicit degree upper bound (Theorem 7.11). We then use simple facts from algebraic number theory to prove that a matrix whose entries are primitive roots of sufficiently high orders cannot satisfy any polynomial with such a degree upper bound. This gives us the claimed family of matrices of maximum rigidity.

Our primary objects of interest in this chapter are the varieties of matrices with rigidity at most k. For a fixed k, we have a natural decomposition of the variety based on the patterns of changes. We prove that this natural decomposition is indeed a decomposition into irreducible components (Corollary 7.17). In fact, these components are defined by elimination ideals of determinantal ideals generated by all the $(r+1) \times (r+1)$ minors of an $n \times n$ matrix of indeterminates. While determinantal ideals have been well-studied in mathematical literature, their elimination theory does not seem to have been studied. Application to rigidity of these elimination ideals of determinantal ideals might be a natural motivation for studying them.

We next consider the question: given a matrix A, is there a small neighbourhood of A within which the rigidity function is nondecreasing, i.e. such that every matrix in this neighbourhood has rigidity at least equal to that of A? This is related to the notion of *semicontinuity* of the rigidity function. We give a family of examples to show that the rigidity function is in general not semicontinuous. However, the *specific* matrices we produce above, by their very construction, have neighbourhoods within which rigidity is nondecreasing.

The rest of the chapter is organised as follows. In the next two subsections, we introduce some definitions and notations, recall a basic result from elimination theory. Much of the necessary background from algebraic geometry is briefly introduced in the appendix B. We introduce our main approach in Section 7.3, reprove Valiant's theorem, and compute the dimension of the variety of matrices of rigidity at most k. We present our new construction of maximally rigid matrices in Section 7.3.3. Connection to the elimination ideals of determinantal ideals is established in Section 7.4. In Section 7.5, we study semicontinuity of the rigidity function through examples and counterexamples.

7.2 Notations & Background

First we recall some notation introduced before. Let $\mathbb{F}$ be a field. Then, by $M_n(\mathbb{F})$ we denote the algebra of $n \times n$ matrices over $\mathbb{F}$. At times, when it is clear from the context, we will denote $M_n(\mathbb{F})$ by M_n. Let $X \in M_n(\mathbb{F})$. Then by X_{ij} we will denote the (i, j)-th entry of X. Given $X \in M_n(F)$, the support of X is defined as $Supp(X) := \{(i, j) \mid X_{ij} \neq 0 \in F\}$. Given a non-negative integer k, we define

$$S(k) := \{X \in M_n(F) : |Supp(X)| \leq k\}.$$

Thus, $S(k)$ is the set of matrices over $\mathbb{F}$ with at most k non-zero entries.

Definition 7.1. *A* pattern π *is a subset of the positions of a matrix of size* $n \times n$*. By* $|\pi|$ *we* *denote the number of non-zero elements in the pattern π.*

For any pattern π, we define:

$$S(\pi) := \{X \in M_n(F) : \ Supp(X) \subseteq \pi\}$$

Note that $S(k) = \cup_{|\pi|=k} S(\pi)$.

We will recall (and reword) the definition of matrix rigidity from the previous chapter, and say that a matrix X is (r, k)-rigid if changing at most k entries of X does not bring down the rank of the matrix to r or less. More formally,

Definition 7.2. *A matrix X is (r, k)-rigid if $rank(X + T) > r$ whenever $T \in S(k)$.*

Definition 7.3. *The rigidity function* $\mathrm{Rig}(X, r)$ *is the smallest integer k for which the matrix X is not (r, k)-rigid. That is,* $\mathrm{Rig}(X, r)$ *is the minimum number of entries we need to change in the matrix X so that the rank becomes at most r:*

$$\mathrm{Rig}(X, r) := \min\{|Supp(T)| \ : \ \mathrm{rank}(X + T) \leq r\}.$$

Sometimes, we will allow T to be chosen in $M_n(L)$ for L an extension field of F. In this case we will denote the rigidity by $Rig(X, r, L)$.

Let $\mathsf{RIG}(n, r, k)$ denote the set of $n \times n$ matrices X such that $\mathrm{Rig}(X, r) = k$. Similarly, we define $\mathsf{RIG}(n, r, \geq k)$ to be the set of matrices of rigidity at least k and $\mathsf{RIG}(n, r, \leq k)$ to be the set of matrices of rigidity at most k. For a pattern π of size k, let $\mathsf{RIG}(n, r, \pi)$ be the set of matrices X such that for some $T \in S(\pi)$ we have $\mathrm{rank}(X + T) \leq r$. Then we have

$$\mathsf{RIG}(n, r, \leq k) = \bigcup_{\pi, |\pi|=k} \mathsf{RIG}(n, r, \pi).$$

7.2.1 Elimination Theory: Closure Theorem

Here we recall a basic result from Elimination Theory. As the name suggests, Elimination Theory deals with elimination of a subset of variables from a given set of polynomial equations and finding the *reduced set* of polynomial equations from which these variables have been eliminated. The main results of Elimination Theory, especially the Closure Theorem, describe a precise relation between the reduced ideal and the given ideal, and its corresponding geometric interpretation. We refer to [CLO07] for a more detailed description.

Definition 7.4 (Elimination Ideal)**.** *Given an ideal $I = \langle f_1, \ldots, f_s \rangle \subseteq F[x_1, \ldots, x_n]$, the l-th* elimination ideal *I_l is the ideal of $F[x_{l+1}, \ldots, x_n]$ defined by*

$$I_l := I \cap F[x_{l+1}, \ldots, x_n].$$

Theorem 7.5 (Closure Theorem)**.** *Let I be an ideal of $F[x_1, \ldots, x_n, y_1, \ldots, y_m]$ and $I_m := I \cap F[y_1, \ldots, y_m]$ be the n-th elimination ideal associated to I. Let $V(I)$ and $V(I_m)$ be the subvarieties of $\mathbb{A}^{n+m}$ and $\mathbb{A}^m$. Let p be the natural projection map from $\mathbb{A}^{n+m} \to \mathbb{A}^m$ (projection map onto the y-coordinates). Then,*

1. *$V(I_n)$ is the smallest (closed) affine variety containing $p(V(I)) \subseteq \mathbb{A}^m$. In other words, $V(I_n)$ is the Zariski closure of $p(V(I))(\overline{\mathbb{F}}) \subseteq \overline{\mathbb{F}}^m$.*
2. *When $V(I)(\overline{\mathbb{F}}) \neq \phi$, there is an affine variety W strictly contained in $V(I_n)$ such that $V(I_n) - W \subseteq p(V(I))$.*

7.3 Use of Elimination Theory

7.3.1 Determinantal Ideals and their Elimination Ideals

We will now investigate the structure of the sets $\mathsf{RIG}(n, r, \leq k)$ and $\mathsf{RIG}(n, r, \pi)$ and their Zariski closures.

$$\begin{aligned} \mathsf{W}(n, r, \leq k) &:= \overline{\mathsf{RIG}(n, r, \leq k)} \\ \mathsf{W}(n, r, \pi) &:= \overline{\mathsf{RIG}(n, r, \pi)} \end{aligned}$$

in the n^2-dimensional affine space of $n \times n$ matrices. Let X be an $n \times n$ matrix with entries being indeterminates $x_1, \ldots, x_{n^2}$. For a pattern π of k positions, let T_π be the $n \times n$ matrix with indeterminates $t_1, \ldots, t_k$ in the positions given by π and zero in the rest. Note that saying $X + T_\pi$ has rank at most r is equivalent to saying that all its $(r+1) \times (r+1)$ minors vanish. Let us consider the ideal generated by these minors:

$$I(n, r, \pi) := \left\langle Minors_{(r+1)\times(r+1)}(X + T_\pi) \right\rangle \subseteq F[x_1, \ldots, x_{n^2}, t_1, \ldots, t_k]. \tag{7.1}$$

It then follows from the definition of rigidity that $\mathsf{RIG}(n, r, \pi)$ is the projection from $\mathbb{A}^{n^2} \times \mathbb{A}^k$ to $\mathbb{A}^{n^2}$ of the algebraic set $V(I(n, r, \pi))(F)$. Thus, if we define the elimination ideal

$$EI(n, r, \pi) := I(n, r, \pi) \cap F[x_1, \ldots, x_{n^2}] \subseteq F[x_1, \ldots, x_{n^2}],$$

then by the Closure Theorem (Theorem 7.5), we obtain

$$\mathsf{W}(n, r, \pi) = V(EI(n, r, \pi)). \tag{7.2}$$

Note that

$$\mathsf{W}(n, r, \leq k) = \bigcup_{\pi, |\pi| = k} \mathsf{W}(n, r, \pi).$$

7.3.2 Valiant's Theorem

The following theorem due to Valiant [Val77, Theorem 6.4, page 172] says that a generic matrix has rigidity $(n-r)^2$. That is, for $k < (n-r)^2$, the dimension of $\mathsf{W}(n, r, \leq k)$ is strictly less than n^2.

In the following we will rephrase Valiant's proof in the language of algebraic geometry. The point of doing this is to set up the formalism and use it later; in particular, when we compute the exact dimension of the rigidity variety $\mathsf{W}(n, r, \leq k)$

Theorem 7.6 (Valiant). *Let $n \geq 1$, $0 < r < n$ and $0 \leq k < (n-r)^2$. Let $\mathsf{W} := \mathsf{W}(n, r, \leq k)$ be as above. Then,*

$$\dim(\mathcal{W}) < n^2.$$

Proof. Let $\pi \subseteq \{(i, j) | 1 \leq i, j \leq n\}$ be a pattern of size k. Let τ be a fixed $r \times r$ minor. Define $\mathsf{RIG}(n, r, \pi, \tau)$ to be the set of all $n \times n$ matrices A that satisfy the following properties: there exists some $n \times n$ matrix T_π such that

1. $Supp(T_\pi) \subseteq \pi$,

2. $\operatorname{rank}(A + T_\pi) = r$, and

3. $\det((A + T_\pi)_\tau) \neq 0$ where τ denotes the fixed $r \times r$ minor as above.

Recall that $S(\pi)$ is the set of matrices whose support is contained in π. Let us also define

$$\mathsf{RANK}(n, r, \tau) := \{C \in M_n \mid \operatorname{rank}(C) = r \text{ and} \det(C_\tau) \neq 0\}.$$

By definition, every element $A \in \mathsf{RIG}(n, r, \pi, \tau)$ can be written as $C + T_\pi$, with $C \in \mathsf{RANK}(n, r, \tau)$ and $T_\pi \in S_\pi$.

Consider the following natural map Φ:

$$\mathbb{A}^{n^2-(n-r)^2} \times \mathbb{A}^k \supset \mathsf{RANK}(n, r, \tau) \times S(\pi) \xrightarrow{\Phi} M_n \cong \mathbb{A}^{n^2}, \tag{7.3}$$

taking (X, T_π) to $X + T_\pi$. The image of Φ is exactly $\mathsf{RIG}(n, r, \pi, \tau)$.

Lemma 7.7.

$$\dim(\mathsf{RANK}(n, r, \tau)) = n^2 - (n - r)^2$$

Proof. Without loss of generality we can assume that the τ is the upper left $r \times r$-minor. Thus we can write a $C \in \mathsf{RANK}(n, r, \tau)$ as

$$C = \begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix},$$

where $\operatorname{rank}(C) = r$ and C_{11} is an $r \times r$ matrix whose determinant is non-zero.

Since the matrix C_{11} is nonsingular of dimension equal to $r = \operatorname{rank}(C)$, it follows that the first r columns are linearly independent and span the column space of C. Therefore each of the last $(n-r)$ columns is a linear combination of the first r columns in exactly one way, and the linear combination is determined by the entries of C_{12}. Formally, we have the equation

$$C_{22} = C_{21} C_{11}^{-1} C_{12}.$$

The set of all C_{11} is an affine open set of dimension r^2 and C_{12} and C_{21} can each range over $\mathbb{A}^{r(n-r)}$. This gives a set of rational $r^2 + 2r(n - r)$ independent rational functions and hence, the algebraic set $\mathsf{RANK}(n, r, \tau)$ has dimension $n^2 - (n - r)^2$. □

Also, note that

$$\dim(S(\pi)) = |\pi|.$$

We note that if there is a surjective morphism from an affine variety X to another affine variety Y, then $\dim Y \leq \dim X$ (a more formal statement appears as Lemma B.5 in Appendix B). Thus for $k \leq (n - r)^2 - 1$, we get

$$\dim(\overline{Im(\Phi)}) = \dim(\overline{\mathsf{RIG}(n, r, \pi, \tau)}) \leq n^2 - (n - r)^2 + k < n^2.$$

Note that

$$\mathsf{W} = \bigcup_{\tau,\pi} \overline{\mathsf{RIG}(n, r, \pi, \tau)}$$

and that completes the proof of the theorem.

□

Thus we have proved that the set of matrices of rigidity strictly smaller than $(n-r)^2$ is contained in a proper closed affine variety of $\mathbb{A}^{n^2}$, and thus is of dimension strictly less than n^2. In other words, a *generic matrix*, i.e. a matrix that lies outside a certain proper closed affine subvariety of $\mathbb{A}^{n^2}$, is *maximally rigid*. Therefore, over an infinite field F (for instance, an algebraically closed field), there always exist maximally rigid matrices.

We now refine Valiant's argument and prove the following exact bound on the dimension of W. The main point of the proof is a *lower bound* on $\dim(\mathsf{W})$.

Theorem 7.8. *Let* $0 \leq r \leq n$ *and* $0 \leq k \leq (n-r)^2$*. Then*

$$\dim(\mathsf{W}) = n^2 - (n-r)^2 + k.$$

Proof. With the notation of the previous proof, we have the map

$$\Phi : \mathsf{RANK}(n, r, \tau) \times S(\pi) \xrightarrow{+} M_n.$$

Let $\mathsf{RANK}(n, \leq r)$ be the set of $n \times n$ matrices of rank at most r. Let $S(k)$ be the set of matrices of support at most k.

Now note that $GL(n) \times GL(n)$ acts on $\mathsf{RANK}(n, \leq r)$ by multiplication on the left and right, and the action is transitive on the set of matrices with rank exactly r, which forms a Zariski open subset of $\mathsf{RANK}(n, \leq r)$. Therefore $\mathsf{RANK}(n, \leq r)$ is an irreducible algebraic variety. It is not difficult to see (for instance, from the computation below of the tangent space; see appendix B for an intuitive explanation) that its singular locus is exactly $\mathsf{RANK}(n, \leq r-1)$, the set of matrices with rank less than r.

On the other hand $S(k)$ splits up into components S_π depending on the pattern π and is thus a union of various affine subspaces (each associated to a π of size at most k). The nonsingular elements of $S(k)$ are those which have support of size exactly k.

We can put together the maps Φ, as defined in the proof of Theorem 7.6, arising from various choices of τ and π to write the map

$$\Phi : \mathsf{RANK}(n, \leq r) \times S(k) \rightarrow \mathsf{RIG}(n, r, \leq k).$$

We have seen that Φ is a surjective morphism of affine varieties. If we can find a nonsingular point of $\mathsf{RANK}(n, \leq r) \times S(k)$ for which the map on tangent spaces is injective, then the dimension of the target space $\mathsf{RIG}(n, r, \leq k)$ will be at least (and hence equal to) $\dim \mathsf{RANK}(n, \leq r) + \dim S(k) = n^2 - (n-r)^2 + k$, proving the theorem. Since the map on tangent spaces is simply addition of matrices, we need to see that the subspaces do not intersect non-trivially and that would complete the proof of the theorem. For any smooth point $x \in \mathsf{RANK}(n, r)$, the smooth locus of $\mathsf{RANK}(n, \leq r)$, we will find a pattern π of size k and $y \in S_\pi$ for which the tangent spaces at x and y intersect transversely.

Assume first that the point x is

$$\begin{pmatrix} I_r & 0 \\ 0 & 0 \end{pmatrix}.$$

We choose the pattern π to lie completely in the bottom right hand block of size $(n-r) \times (n-r)$, and choose any smooth point y of S_π (i.e. having all k entries nonzero).

The tangent space of x is

$$\begin{pmatrix} * & * \\ * & 0 \end{pmatrix}.$$

That is, it consists of the subspace of M_n consisting of matrices with arbitrary entries except in the lower $(n-r) \times (n-r)$ block, which is constrained to be the zero submatrix. The dimension of the tangent space is $r^2 + 2r(n-r) = n^2 - (n-r)^2$, as expected.

The tangent space of y is

$$\begin{pmatrix} 0 & 0 \\ 0 & *_\pi \end{pmatrix}$$

where $*_\pi$ means that the entries in positions of π are arbitrary, and the other entries are zero.

It's clear that the two tangent spaces intersect transversely. Now we need to show this for more a general $x \in \mathsf{RANK}(n, r)$. Assume that the top left $r \times r$ minor of x is nonsingular (else we can multiply by permutation matrices on left and right, noting that permutations just shuffle the various $S(\pi)$ for $|\pi| = k$).

The first r columns of x are independent and span the column space of x, so there exists a matrix

$$g = \begin{pmatrix} I_r & * \\ 0 & I_{n-r} \end{pmatrix}$$

such that xg has the form

$$\begin{pmatrix} * & 0 \\ * & 0 \end{pmatrix}.$$

Then using that the first r rows of xg are independent and span its row space, we can find an invertible matrix

$$h = \begin{pmatrix} * & 0 \\ * & I_{n-r} \end{pmatrix}$$

such that

$$hxg = \begin{pmatrix} I_{n-r} & 0 \\ 0 & 0 \end{pmatrix}.$$

The tangent space of x is

$$h^{-1} \begin{pmatrix} * & * \\ * & 0 \end{pmatrix} g^{-1}.$$

We need to show this does not intersect $S(\pi)$ for some π. That is,

$$\begin{pmatrix} * & * \\ * & 0 \end{pmatrix}$$

does not intersect

$$h \begin{pmatrix} 0 & 0 \\ 0 & *_\pi \end{pmatrix} g$$

except in zero. But this follows from the fact that the latter is a matrix of the same form (in fact, multiplication by h and g leave any element of $S(\pi)$ unchanged). □

Remark 7.9. *A similar line of study - though in the projective setting - is found in [LTV03]. However, we think that our formalism and proofs are clearer and simpler, and gives an explicit bound on the dimension of the set of matrices considered.*

7.3.3 Rigid Matrices over $\mathbb{C}$

Recall that to say that the rigidity of a matrix A for target rank r is at least k it suffices to prove that the matrix A is not in $\mathsf{W}(n, r, \leq (k-1))$. We use this idea to achieve the maximum possible lower bound for the rigidity of a family of matrices over the field of complex numbers $\mathbb{C}$. As a matter of fact, we obtain matrices with real algebraic entries with rigidity $(n-r)^2$.

Theorem 7.10. *Let $\delta(n) = n^{4n^4}$. Let $p_{i,j}$ for $1 \leq i,\ j \leq n$ be distinct primes such that $p_{i,j} > \delta(n)$. Let $K = \mathbb{Q}(\zeta_{1,1}, \ldots, \zeta_{n,n})$ where $\zeta_{i,j} = e^{2\pi \mathrm{i}/p_{i,j}}$. Let $A(n) := (\zeta_{i,j}) \in M(n, K)$. Then, for any field L containing K,*

$$\mathrm{Rig}(A(n), r, L) = (n - r)^2.$$

Proof. For simplicity, we will index the $\zeta_{i,j}$ by ζ_α for $\alpha = 1$ to n^2, and similarly p_α. We prove the theorem by showing that

$$A(n) \notin \mathsf{W}(n, r, \leq (n - r)^2 - 1)(L).$$

Thus it is sufficient to prove that

$$A(n) \notin \mathsf{W}(n, r, \pi)(L)$$

for any pattern π with $|\pi| = (n - r)^2 - 1$. Let π be any such pattern. To simplify notation, let us define, $\mathsf{W} := \mathsf{W}(n, r, \pi)(L)$. By Theorem(7.6) we have:

$$\dim(\mathsf{W}) \leq \dim(\mathsf{W}(n, r, \leq (n - r)^2 - 1)) \leq (n^2 - 1) < n^2$$

Equivalently (by Hilbert's Nullstellensatz),

$$EI(n, r, \pi) \neq (0).$$

Proving that $A(n) \notin \mathsf{W}$ is equivalent to showing the existence of a $g \in EI(n, r, \pi)$ such that $g(A(n)) \neq 0$. We produce such a g using the following theorem:

Theorem 7.11. *([BMMR02, Page 6,Theorem 4]) Let $I = \langle f_1, \ldots, f_s \rangle$ be an ideal in the polynomial ring $\mathbb{F}[Y]$ over an infinite field $\mathbb{F}$, where $Y = \{y_1, \ldots, y_m\}$. Let d be the maximum* total degree *of the generators f_i. Let $Z = \{y_{i_1}, \ldots, y_{i_\ell}\} \subseteq Y$ be a subset of indeterminates of Y. If $I \cap F[Z] \neq (0)$ then there exists a non-zero polynomial $g \in I \cap F[Z]$ such that, $g = \sum_{i=1}^{s} g_i f_i$, with $g_i \in F[Y]$ and*

$$\deg(g_i f_i) \leq (\mu + 1)(m + 2)(d^\mu + 1)^{\mu+2},$$

where $\mu = min\{s, m\}$.

Let us apply Theorem(7.11) to our case - in the notation of this theorem our data is as follows:

$$\begin{aligned}
\mathbb{F} &:= \mathbb{Q} \\
Y &:= \{x_1, \ldots, x_{n^2}, t_1, \ldots, t_k\} \\
Z &:= \{x_1, \ldots, x_{n^2}\} \\
\Sigma_{r+1} &:= \text{set of all minors of size } (r+1) \\
f_\tau &:= \det((X + T_\pi)_\tau) \text{ for } \tau \in \Sigma_{r+1}
\end{aligned}$$

Here by Y_τ we denote the τ-th minor of Y, and $I := I(n, r, \pi) = \langle f_\tau : \tau \in \Sigma_{r+1} \rangle$ as in (7.1).

Furthermore, we have:

$$\begin{aligned}
m &= n^2 + (n-r)^2 - 1 \leq 2n^2 - 2 \\
\mu &= \min\left\{ n^2 + (n-r)^2 - 1, \binom{n}{r+1}^2 \right\} \\
&\leq n^2 + (n-r)^2 - 1 \leq 2n^2 - 2, \\
d &= r + 1 \leq n, \\
I \cap F[Z] &= EI(n, r, \pi) \neq (0).
\end{aligned}$$

By Theorem(7.11) there exists a

$$g \neq 0 \in EI(n, r, \pi) \subseteq \mathbb{Q}[x_1, \ldots, x_{n^2}]$$

such that

$$\deg(g) \leq (2n^2 - 1)(2n^2)(n^{2n^2-2} + 1)^{2n^2} < n^{4n^4} = \delta(n).$$

Without loss of generality, let x_1 be a variable that appears in $g(x_1, \ldots, x_n)$. Let $l := \deg_{x_1}(g) < N$. Thus,

$$g(\underline{x}) = \sum_{i=0}^{l} f_i(x_2, \ldots, x_{n^2})\, x_1^i, \text{ where } f_i \in \mathbb{Q}[x_2, \ldots, x_{n^2}].$$

We will now apply the following Lemma 7.12 to this situation.

Lemma 7.12. *Let N be a positive integer. Let $\theta_1, \ldots, \theta_m$ be m algebraic numbers such that* *for any $1 \leq i \leq m$, the field $\mathbb{Q}(\theta_i)$ is Galois over $\mathbb{Q}$ and such that*

$$[\mathbb{Q}(\theta_i) : \mathbb{Q}] \geq N,$$

$$\mathbb{Q}(\theta_i) \cap \mathbb{Q}(\theta_1, \ldots, \theta_{i-1}, \theta_{i+1}, \ldots, \theta_m) = \mathbb{Q}.$$

Let $g(\underline{x}) \neq 0 \in \mathbb{Q}[x_1, \ldots, x_m]$ such that $\deg(g) < N$. Then,

$$g(\theta_1, \ldots, \theta_m) \neq 0.$$

Let us set $m = n^2$, $N = \delta(n)$, $l := \deg(g) \leq N$ in Lemma(7.12). We need to ensure two conditions to apply the above lemma. The first condition:

$$[\mathbb{Q}(\zeta_\alpha) : \mathbb{Q}] = p_\alpha - 1 \geq \delta(n) = N$$

follows simply because the minimal polynomial of ζ_α is $x^{p_\alpha - 1} - 1$. The second one follows from standard facts in number theory:

Claim 1 (See Theorem 4.10 in [Nar04])**.**

$$\mathbb{Q}(\zeta_\alpha) \cap \mathbb{Q}(\zeta_1, \ldots, \zeta_{\alpha-1}, \zeta_{\alpha+1}, \ldots, \zeta_{n^2}) = \mathbb{Q}.$$

To give a little intuition here: the prime p_α is totally ramified in $\mathbb{Q}(\zeta_\alpha)$ (this means that the ideal generated by p_α in the ring of integers of the number field, factors into one single prime ideal, see the appendix C). At the same time, the prime p_α is totally unramified in $\mathbb{Q}(\zeta_1, \ldots, \zeta_{\alpha-1}, \zeta_{\alpha+1}, \ldots, \zeta_{n^2})$.

Thus, Lemma(7.12) is applicable and we get:

$$g(\zeta_1, \ldots, \zeta_{n^2}) \neq 0$$

and that completes the proof of Theorem(7.10). □

Note that Theorem(7.10) is true for any family of matrices $A(n) = [\theta_{i,j}]$ provided the $\theta_{i,j}$ satisfy Lemma(7.12). Thus, we have:

Corollary 7.13 (Rigid Matrices over $\mathbb{R}$)**.** *Let $A(n) := (\zeta_{i,j} + \overline{\zeta_{i,j}})$, where $\zeta_{i,j}$ are primitive roots of unity of order $p_{i,j}$ such that $p_{i,j} - 1 \geq 2\delta(n)$ (here $\overline{\zeta_{i,j}}$ denotes the complex conjugate of $\zeta_{i,j}$). Then, $A(n) \in M(n, \mathbb{R})$ has $\mathrm{Rig}(A(n), r) = (n-r)^2$.*

Proof of Lemma 7.12: By induction on m. For $m = 1$ this is trivial.

Now suppose that the statement is true when the number of variables is strictly less than m. Assuming that the statement is not true for m, we will arrive at a contradiction. This will prove the Lemma.

Let $g \in \mathbb{Q}[\underline{x}]$ with $l := \deg(g) < N$ be such that

$$g(\theta_1, \ldots, \theta_m) = 0,$$

with θ_i, $1 \leq i \leq m$, satisfying the conditions as in the theorem. Since the statement is true for any $(m-1)$ number of variables, without loss of generality, we can assume that all the variables and hence x_m appears in g. Let us denote x_m by x. Let us write

$$g(x_1, \ldots, x_m) = \sum_{i=0}^{l} f_i(x_1, \ldots, x_{m-1}) x^{l-i}.$$

Note that $l < N$ and $\deg(f_i) < N$ for $0 \leq i \leq l$. Since $g \neq 0$, for some $i, \ 0 \leq i \leq l$ the polynomial $f_i \neq 0$. Thus, by the inductive hypothesis,

$$f_i(\theta_1, \ldots, \theta_{m-1}) \neq 0.$$

Thus $g(\theta_1, \ldots, \theta_{m-1})(x) \neq 0 \in \mathbb{Q}(\theta_1, \ldots, \theta_{m-1})[x]$. This implies that θ_m satisfies a non-zero polynomial over $\mathbb{Q}(\theta_1, \ldots, \theta_{m-1})$ of degree $l < N$. Thus:

$$[\mathbb{Q}(\theta_1, \ldots, \theta_m) : \mathbb{Q}(\theta_1, \ldots, \theta_{m-1})] \leq l < N. \tag{7.4}$$

On the other hand, since $\mathbb{Q}(\theta_m) \cap \mathbb{Q}(\theta_1, \ldots, \theta_{m-1}) = \mathbb{Q}$ and the fields $\mathbb{Q}(\theta_i)$ are Galois over $\mathbb{Q}$, by Theorem(7.14) (stated below), we conclude that

$$[\mathbb{Q}(\theta_1, \ldots, \theta_{m-1})(\theta_m) : \mathbb{Q}(\theta_1, \ldots, \theta_{m-1})] = [\mathbb{Q}(\theta_m) : \mathbb{Q}] > N$$

This contradicts (7.4) above and that proves the lemma.

For two field $\mathbb{K}$ and $\mathbb{F}$, let $\mathbb{KF}$ denote the compositum of the two fields, which is the smallest field containing both $\mathbb{K}$ and $\mathbb{F}$.

Theorem 7.14 ([Lan04], Theorem 1.12, page 266). *Let $\mathbb{K}$ be a Galois extension of k, let $\mathbb{F}$ be an arbitrary extension and assume that $\mathbb{K}$, $\mathbb{F}$ are subfields of some other field L. Then $\mathbb{KF}$ (the composition of $\mathbb{K}$ and $\mathbb{F}$) is Galois over $\mathbb{F}$, and $\mathbb{K}$ is Galois over $K \bigcap F$. Let H be the Galois group of $\mathbb{KF}$ over $\mathbb{F}$, and G the Galois group of $\mathbb{K}$ over k. If $\sigma \in H$ then the restriction* *of σ to $\mathbb{K}$ is in G, and the map:*

$$\sigma \mapsto \sigma|_K$$

gives an isomorphism of H on the Galois group of K over $K \cap F$. In particular,

$$[KF : K] = [K : K \cap F].$$

□

7.4 Reduction to Determinantal Ideals

This section shows that, in order to prove bounds on rigidity, it is sufficient to address the question for the class of determinantal ideals associated with it. This provides a simplification and it may be possible that better bounds are known for such kind of ideals.

More precisely, we show that the natural decomposition of the rigidity variety $\mathsf{W}(n, r, \leq k) = \cup_{|\pi|=k}\mathsf{W}(n, r, \pi)$ is indeed a decomposition into irreducible affine algebraic varieties. In fact, these components turn out to be varieties defined by elimination ideals of determinantal ideals generated by all the $(r + 1) \times (r + 1)$ minors.

We will continue to use the notation from Section 7.3. Consider the matrix $X + T_\pi$. Let $x = \{x_1, \ldots, x_{n^2}\} = x_{\bar{\pi}} \cup x_\pi$, where x_π is the set of variables that are indexed by π and $x_{\bar{\pi}}$ is the set of remaining variables.

Let

$$J := I(n, r, \pi) = \langle Minors_{(r+1)\times(r+1)}(X + T_\pi)\rangle$$

be the ideal of $\mathbb{Q}[x, t] = \mathbb{Q}[x_\pi, x_{\bar{\pi}}, t_\pi]$ generated by the $(r + 1) \times (r + 1)$ minors of $X + T_\pi$. Let

$$\begin{aligned} J_1 &:= J \cap \mathbb{Q}[x_\pi, x_{\bar{\pi}}] \subseteq \mathbb{Q}[x_1, \ldots, x_{n^2}] \\ J_2 &:= J_1 \cap \mathbb{Q}[x_{\bar{\pi}}] \\ I_{r+1} &:= \langle Minors_{(r+1)\times(r+1)}(X)\rangle \subseteq \mathbb{Q}[x] \\ EI_{r+1} &:= I_{r+1} \cap \mathbb{Q}[x_{\bar{\pi}}] \subseteq \mathbb{Q}[x_{\bar{\pi}}] \end{aligned}$$

Notice that since J_1 is the elimination ideal of J w.r.t. eliminating variables t_π, a matrix A lies in $\mathsf{W}(n, r, \leq k) = \overline{\mathsf{RIG}(n, r, \leq k)}$ if and only if its entries lie in the variety defined by the ideal J_1. Also, I_{r+1} is the ideal generated by the $(r+1)\times(r+1)$ minors of X and EI_{r+1} its elimination ideal for the rational ring generated by the variables $x_{\bar{\pi}}$.

Proposition 7.15. *$J_1 = J_2\mathbb{Q}[x]$ (the ideal generated by J_2 in $\mathbb{Q}[x]$) and $J_2 = EI_{r+1}$. In particular, $EI(n, r, \pi) = EI_{r+1}\mathbb{Q}[x]$ considered as ideals in $\mathbb{Q}[x]$.*

Proof. First, notice that in the $(r + 1) \times (r + 1)$ minors of $X + T_\pi$, the variable $t_{i,j}$, for $(i, j) \in \pi$, always occurs in combination with $x_{i,j}$ as $t_{i,j} + x_{i,j}$. Therefore, eliminating the variables t_π will also automatically eliminate the variables x_π, giving the equality of the generators of the ideals J_1 and J_2. Therefore $J_1 = J_2\mathbb{Q}[x]$. More formally, consider the isomorphism between the two coordinate rings $\phi : \mathbb{Q}[x_\pi, x_{\bar{\pi}}, t_\pi]$ and $\mathbb{Q}[x_\pi, x_{\bar{\pi}}, t_\pi]$ defined by letting $\phi(t_{i,j}) = x_{i,j} + t_{i,j}$ for each $(i, j) \in \pi$ and $\phi(x_{i,j}) = x_{i,j}$ for all $(i, j) \notin \pi$. The ideal $J_1 = J \cap \mathbb{Q}[x_\pi, x_{\bar{\pi}}] \subseteq \mathbb{Q}[x_1, \ldots, x_{n^2}]$ must equal the ideal $\phi(\phi^{-1}(J) \cap \phi^{-1}\mathbb{Q}[x_1, \ldots, x_{n^2}])$, since ϕ is an isomorphism. But $\phi^{-1}(J)$ is generated by matrices only involving the variables of t_π and $x_{\bar{\pi}}$, whereas $\phi^{-1}\mathbb{Q}[x_1, \ldots, x_{n^2}]) = \mathbb{Q}[x_1, \ldots, x_{n^2}]$, so that $\phi^{-1}(J) \cap \phi^{-1}\mathbb{Q}[x_1, \ldots, x_{n^2}]$ is generated by polynomials only involving the variables of $x_{\bar{\pi}}$. Therefore $\phi^{-1}(J_1) = \phi^{-1}(J) \cap \phi^{-1}\mathbb{Q}[x_1, \ldots, x_{n^2}] = J_2\mathbb{Q}[x]$ and taking the image under ϕ, we get $J_1 = J_2\mathbb{Q}[x]$.

The equation $J_2 = EI_{r+1}$ follows from similar considerations, noting that the variables $x_{i,j}$ for $(i, j) \in \pi$ always occur in the combination $x_{i,j} + t_{i,j}$. Therefore eliminating them eliminates $t_{i,j}$ as well. More formally, consider the isomorphism $\psi : \mathbb{Q}[x_\pi, x_{\bar{\pi}}, t_\pi] \to \mathbb{Q}[x_\pi, x_{\bar{\pi}}, t_\pi]$ defined by letting $\psi(x_{i,j}) = x_{i,j} + t_{i,j}$ for each $(i, j) \in \pi$, while $\psi(t_{i,j}) = t_{i,j}$ for $(i, j) \in \pi$ and $\psi(x_{i,j}) = x_{i,j}$ for $(i, j) \notin \pi$. Then again we have $J_2 = J_1 \cap \mathbb{Q}[x_{\bar{\pi}}] = J \cap \mathbb{Q}[x_{\bar{\pi}}] = \psi(\psi^{-1}(J) \cap \psi^{-1}(\mathbb{Q}[x_{\bar{\pi}}])) = \phi(I_{r+1}Q[x, t_\pi] \cap Q[x_{\bar{\pi}}]) = \phi(EI_{r+1}) = EI_{r+1} \subset Q[x_{\bar{\pi}}]$. □

The following is a well-known theorem; see [HE71, Theorem 1] and [BV80, Chapter 2].

Theorem 7.16. *Let $\mathsf{RANK}(n, \leq r)$ be the set of all rank $\leq r$ matrices of $M_n \cong \mathbb{A}^{n^2}$. Then*

- *$I(\mathsf{RANK}(n, \leq r)) = I_{r+1}$ and $\mathsf{RANK}(n, \leq r) = V(I_{r+1})$.*
- *I_{r+1} is a prime ideal of $\mathbb{Q}[X]$. In particular, $\mathsf{RANK}(n, \leq r)$ is an irreducible variety.*

Corollary 7.17. *In the natural decomposition $\mathsf{W}(n, r, \leq k) = \cup_{|\pi|=k}\mathsf{W}(n, r, \pi)$, the $\mathsf{W}(n, r, \pi)$ are irreducible varieties.*

Proof. The elimination ideal $EI_{r+1} \subseteq \mathbb{Q}[x_{\bar{\pi}}]$ is a prime ideal since $I_{r+1} \subseteq \mathbb{Q}[x]$ is prime by Theorem 7.16. By Proposition 7.15, $V(EI_{r+1}) = V(EI(n, r, \pi))$ is irreducible in $\mathbb{A}^{n^2}$. Now, by (7.2), we conclude that $\mathsf{W}(n, r, \pi)$ is an irreducible affine variety. □

7.5 Semi-continuity of Rigidity

Currently we know ([Lok00], Theorem 7.10, Corollary 7.10) families of matrices lower bounds for matrices whose entries are in $\mathbb{C}$ (and over $\mathbb{R}$). One possible approach to get to an explicit family of matrices, from these known families is to search in their neighbourhoods (in the Euclidean topology) to get to a rational matrix. An effectively computable realization of this approach demands for an explicit $\mathsf{poly}(n)$ bound on the number of bits needed to represent each entry of the $n \times n$ matrix.

However, this indeed motivates the study of continuity properties of rigidity. Intuitively, if a function is (lower) semi-continuous at a given point, then within a small neighbourhood of that point the function is nondecreasing. More formally,

Definition 7.18 (Semi-continuity). *Let Y be a topological space. A function $\phi : Y \to Z$ is lower semi-continuous if for each ℓ, the set $\{y \in Y \mid \phi(y) \leq \ell\}$ is a closed subset of Y. That is, for each y there is a neighbourhood U of y such that for $y' \in U, \phi(y') \geq \phi(y)$. Intuitively, ϕ can jump up only at special points, and it can't jump down.*

In contrast, if for each y there is a neighbourhood U of y such that for $y' \in U, \phi(y') \leq \phi(y)$, then we say that the function ϕ is upper semi-continuous.

In our context, we will be interested in studying lower semi-continuity properties of the rigidity function. To make this a little more precise, let Y be a topological space and let $\psi : Y \to \mathbb{R}^{n\times n}$ be continuous. Then we want to study the continuity properties of the function $\mathrm{Rig}_r : y \mapsto Rig(\psi(y), r)$.

We start with a simple case; namely the rank function. The rank function of a matrix, for example, is a lower semi-continuous function on the space of all $n \times n$ complex matrices as shown in the following proposition (This follows from basic ideas, but somewhat detailed description appears in [Lew06].)

Proposition 7.19 (Lower semi-continuity of rank). *Let Y be a topological space and let $\psi : Y \to \mathbb{R}^{n\times n}$ be continuous, then the function $\phi : y \mapsto \mathrm{rank}(\psi(y))$ is lower semi-continuous.*

Proof. For any y, if $\mathrm{rank}(\psi(y)) = r$, there is an $r \times r$ sub-determinant which becomes nonzero at $\psi(y)$. Let this determinant be the multi-linear polynomial p, and we know that $p(\psi(y)) \neq 0$. However, since polynomials are continuous there is a small neighbourhood U_y of y such that, for all $u \in U$ $p(u) \neq 0$. Thus for all $u \in U$, $\mathrm{rank}(u) \geq r$. In other words, if A in a $n \times n$ matrix over $\mathbb{C}$ and $rank(A) = r$ then there is an $\epsilon = \epsilon(A, k, r)$ such that for all matrices B in the ϵ ball of A we have $\mathrm{rank}(B) \geq r$.

In fact, the same proof also shows that nullity function is upper semi-continuous. □

Upper Semi-continuity: We study this for completeness, although it does not suit for our lower bound applications. The upper semi-continuity property does not hold for rigidity function. We give a direct counter example. Let A be the $n \times n$ matrix obtained by adjoining identity matrix I_m, $r < m < n$ with the 0 matrix, and let r be the target rank. Clearly $R_M(r) \leq m - r$. Now for any ϵ-ball U around M, construct the a matrix M' in U as follows: put the ϵ on the diagonal of the 0-matrix in M to get a matrix M' such that $R_{M'}(r) > m - r$.

Lower Semi-continuity: From now on we look at only lower semi-continuity, which is more relevant for our applications. Unless otherwise mentioned, by semi-continuity we mean lower semi-continuity.

To begin with, we see that the above lemma shows the semi-continuity of rigidity function for a restricted case of $\mathsf{RIG}(M, r, k)$, where $k = 1$. However, rigidity function does not have semi-continuity properties in general. In this section, we give examples to show that the rigidity function is not semicontinuous in general. However, it seems to have semi-continuity property at some interesting matrices. In particular, the matrices $A(n)$ from Theorem 7.10 and Corollary 7.13 have an open neighbourhood around them within which the rigidity function is constant. This is a direct consequence of their very construction since they are outside the closed sets $\mathsf{W}(n, r, \leq (n-r)^2 - 1)$. Another finite example with square roots of primes as its entries appears in below. However, we do not know if the matrices produced in [Lok00](where the (i, j) entry of the matrix is the square root of $\sqrt{p_{ij}}$ for distinct primes p_{ij}, see (11) of table 6.1) have this property.

We illustrate that rigidity function is not lower semi-continuous. That is, we show that there is an infinite family of matrices $\{M_n\}_{n \geq 1}$, for which for any n, for any ϵ_n, there is a matrix N_n which is ϵ_n-close to M_n such that rigidity of N_n is strictly smaller than that of M_n.

The following is an example for 3×3 matrices. Let a, b, c, d, e be non-zero rational numbers. Consider

$$A = \begin{bmatrix} a & b & c \\ d & 0 & 0 \\ e & 0 & 0 \end{bmatrix} \in M(3, \mathbb{C})$$

Observe that $rank(A) = 2$ and by changing two entries its rank can be brought down to 1.

Hence, $\mathrm{Rig}(A, 1) = 2$. Now for any $\epsilon > 0$ let

$$B = \begin{bmatrix} a & b & c \\ d & bd\delta & cd\delta \\ e & be\delta & ce\delta \end{bmatrix} \in M(3, \mathbb{C}),$$

where $\delta \neq 0$ and $\delta \neq 1/a$ is such that $\epsilon \geq \max\{bd\delta, cd\delta, be\delta, ce\delta\}$. Thus B is in the open ball of radius ϵ around A. Note that $rank(B) = 2$. Also $\mathrm{Rig}(B, 1) = 1$ because changing a to $\frac{1}{\delta}$ will make all the 2×2 sub-determinants of B zero. Thus, we have a matrix B which is in the ϵ open ball around A such that $\mathrm{Rig}(A, 1) > \mathrm{Rig}(B, 1)$. To produce an infinite family, for any given n, let

$$A_n := \begin{bmatrix} \alpha & a_1 & a_2 & \dots & a_{n-1} \\ b_1 & 0 & 0 & \dots & 0 \\ b_2 & 0 & 0 & \dots & 0 \\ . & . & . & \dots & . \\ . & . & . & \dots & . \\ b_{n-1} & 0 & 0 & \dots & 0 \end{bmatrix} \in M(n, \mathbb{C}).$$

Then, we have:

Lemma 7.20. *For $n \geq 3$, $rank(A_n) = 2$, $\mathrm{Rig}(A_n, 1) = n - 1$.*

Proof. By Induction: We already argued for the base case when $n = 3$.

Let $n \geq 3$, if we expand the determinant by the first row, every minor is of the form A_{n-1}, and hence have the determinants zero by induction. Thus the rank is same as the rank of A_3 which is 2. It is easy to see that $\mathrm{rank}(A_n) = 2$. In fact, all the 2×2 subdeterminants involving a_i, b_i and α are non-zero. So we have to change at least $(n - 1)$ entries so that all the 2×2 subdeterminants vanish. On the other hand, it suffices to change all the a_i from $i = 2$ to n to reduce the rank to 1. □

Similarly for any ϵ, choose an δ such that $\epsilon \geq \max_{i,j}\{a_i b_j \delta\}$.

$$B_n = \begin{bmatrix} \alpha & a_1 & a_2 & \dots & a_n \\ b_1 & a_1 b_1 \delta & a_2 b_1 \delta & \dots & a_n b_1 \delta \\ b_2 & a_1 b_2 \delta & a_2 b_2 \delta & \dots & a_n b_2 \delta \\ . & . & . & \dots & . \\ . & . & . & \dots & . \\ b_n & a_1 b_n \delta & a_2 b_n \delta & \dots & a_n b_n \delta \end{bmatrix} \in M(n, \mathbb{C})$$

Observe that for every sub-determinant of A_n that is non-zero, the corresponding sub-determinant of B_n will also remain non-zero. Thus $rank(B_n) = 2$. Also $\mathrm{Rig}(B_n, 1) = 1$ because if one changes α to $\frac{1}{\delta}$ then every 2×2 sub-determinant becomes zero. Now we concentrate more on the 3×3 example A_3

$$A = \begin{bmatrix} a & b & c \\ d & 0 & 0 \\ e & 0 & 0 \end{bmatrix}$$

As seen earlier, $A \in \mathsf{RIG}(3, 1, 2)$ and yet there are matrices arbitrarily close to it that belong to $\mathsf{RIG}(3, 1, 1)$. Thus A is in the Euclidean closure of $\mathsf{RIG}(3, 1, 1)$, hence it is also in the Zariski closure of $\mathsf{RIG}(3, 1, 1)$, since the Euclidean or complex topology is finer than the Zariski topology.

Let us verify this directly. We want to verify that $A \in \bigcup_\pi \mathsf{W}(3, 1, \pi, \leq 1)$. We do this by demonstrating a pattern π such that $A \in \mathsf{W}(3, 1, \pi, \leq 1)$. Let $\pi := \{(1, 1)\}$. Let us write:

$$X + t_1 := \begin{bmatrix} x_1 + t_1 & x_2 & x_3 \\ x_3 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{bmatrix}$$

where t_1 is the variable associate to π. Here we get:

$$\begin{aligned} I(3, 1, 1, \pi) \quad = \quad & \langle t_1 x_5 + x_1 x_5 - x_2 x_4, t_1 x_6 + x_1 x_6 - x_3 x_4, \\ & t_1 x_8 + x_1 x_8 - x_2 x_7, t_1 x_9 + x_1 x_9 - x_3 x_7, \\ & x_2 x_6 - x_3 x_5, x_2 x_9 - x_3 x_8, x_4 x_8 - x_5 x_7, \\ & x_4 x_9 - x_6 x_7, x_5 x_9 - x_6 x_8 \rangle \end{aligned}$$

Eliminating t_1 from $I(3,1,1,\pi)$ using the Groebner Basis algorithm we get:

$$\begin{aligned} EI(3,1,1,\pi) &= \langle x_2x_6 - x_3x_5, x_2x_9 - x_3x_8, x_4x_8 - x_5x_7, \\ &\quad x_4x_9 - x_6x_7, x_5x_9 - x_6x_8 \rangle \end{aligned}$$

Note that A does satisfy these generating polynomials. However, this does not mean that A is in the Euclidean closure, as in general, it could be that Euclidean closure is strictly smaller than the Zariski closure.

Examples which are maximally rigid: Now we produce examples of matrices with maximum rigidity where the semi-continuity property of rigidity fails. Take a matrix

$$A = \begin{bmatrix} a & b & c \\ d & e & 0 \\ g & 0 & i \end{bmatrix}$$

where $a, b, \ldots, g$ are non-zero rational numbers. $n = 3, r = 1, k = 3$. Notice that changing 4 entries (namely a, b, d, e) will be enough to bring the rank down to 1. It is easy to verify that changing 3 entries will not suffice for a general choice of $a, \ldots, i$. Thus, $\mathrm{Rig}(A, 1) = 4 = (3-1)^2 = (n-r)^2$.

Let M be a generic matrix and let π be the diagonal pattern of size 3 (represented by variables t_1, t_2, t_3. Consider:

$$M + T_\pi = \begin{bmatrix} a + t_1 & b & c \\ d & e + t_2 & f \\ g & h & i + t_3 \end{bmatrix}$$

It can be checked that the elimination ideal is generated by $bfg - cdh$. Note that A satisfies this equation and thus it follows that $A \in \overline{\mathsf{RIG}(3,1,3,\pi)}$. This implies that any Zariski open neighbourhood of A intersects $\mathsf{RIG}(3,1,3,\pi)$. This is straightforward consequence of the definitions. *What is unclear is whether every Euclidean neighbourhood of such an A intersects* $\mathsf{RIG}(3,1,3,\pi)$.

A Technique for proving semicontinuity: However, this suggests a technique for proving that there is an ϵ such that ϵ-neighbourhood of a matrix does not contain matrices of strictly smaller rigidity. For this we closely studied the Zariski closure of matrices of rigidity at most $k-1$ (for some k). For a matrix M of rigidity at least k, if we prove that it does not lie in the above closure, it means that it is in the complement of a Zariski closed set, and hence in a Euclidean open set. Thus there is an ϵ such that ϵ-neighbourhood of M matrix does not contain matrices of rigidity smaller than k.

We illustrate the above technique by an example: Consider the matrix

$$M := \begin{bmatrix} \sqrt{2} & \sqrt{3} & \sqrt{5} \\ \sqrt{7} & \sqrt{11} & \sqrt{13} \\ \sqrt{17} & \sqrt{19} & \sqrt{23} \end{bmatrix} \in M(3, \mathbb{C}).$$

It is easy to check that $\mathrm{Rig}(M, 1) = 4$. That is $M \in \mathsf{RIG}(3, 1, 4)$, and we want to prove that $M \notin \mathsf{W}(3, 1, 3))$.

We want to check this for all patterns π. But we can rule out some of them quickly as follows. Consider the pattern matrix T_π such that

$$M + T_\pi = \begin{bmatrix} a+t_1 & b+t_2 & c+t_3 \\ d & e & f \\ g & h & i \end{bmatrix}$$

In the elimination ideal, the equation: $\begin{vmatrix} e & f \\ h & i \end{vmatrix} = 0$ which will not be satisfied by M. It is easy to check that the matrix M, due to its choice of entries, has the property that all the submatrices have full rank. Thus, the pattern T_π should touch all 2×2 minors. Thus, up to permutations (since choice of primes in M could be arbitrary but distinct) we need to check the case when T_π has the variables on the diagonal

$$M + T_\pi = \begin{bmatrix} a+t_1 & b & c \\ d & e+t_2 & f \\ g & h & i+t_3 \end{bmatrix}$$

In this case, the elimination ideal is generated by a single polynomial, namely $bfg - cdh$, which M does not satisfy. Since up to permutations, all patterns of size 3 can be written as above, we conclude that $M \notin \mathsf{W}(3, 1, 3)$. But in addition, by the above argument about semi-continuity, it will also imply that for the matrix M_p, there is an ϵ such that all the matrices in the ϵ-neighbourhood are outside $\mathsf{W}(3, 1, 3)$.

$\mathsf{RIG}(n, r, \leq k)(\mathbb{C})$: Euclidean Closure vs Zariski Closure:

In all the examples of the matrices that we constructed we proved that the matrix violates semicontinuity by demonstrating that it lies in the Euclidean closure of matrices of smaller rigidity. Thus they are also in the Zariski closure. It is natural to ask if one can construct examples, where the matrix is outside the Euclidean closure of the matrices of smaller rigidity, but inside their Zariski closure. This leads to the following natural question. How do we compare the two closures of $\mathsf{RIG}(n, r, \leq k)(\mathbb{C})$. We settle this question in the following:

Proposition 7.21. *The Euclidean Closure of* $\mathsf{RIG}(n, r, \leq k)(\mathbb{C})$ *equals its Zariski Closure.*

Proof. Recall that we can write

$$\mathsf{RIG}(n, r, \leq k) = \bigcup_{\pi,\ |\ \pi|=k} \mathsf{RIG}(n, r, \pi).$$

Thus, to prove the proposition, it is sufficient to prove that for any pattern π, the Euclidean closure of $\mathsf{RIG}(n, r, \pi)$ equals its Zariski Closure.

By Closure Theorem, there exists a subvariety V strictly contained in $\mathsf{W} := \mathsf{RIG}(n, r, \pi)$ such that

$$\mathsf{W}(\mathbb{C}) - V(\mathbb{C}) \subseteq \mathsf{RIG}(n, r, \pi)(\mathbb{C}) \subseteq \mathsf{W}(\mathbb{C})$$

Since $\mathsf{W}(\mathbb{C})$ is closed in the Euclidean topology, we will done if we prove that the Euclidean closure of $\mathsf{W}(\mathbb{C}) - V(\mathbb{C})$ is $\mathsf{W}(\mathbb{C})$. This is precisely the statement of the following lemma from [Sha94b], which we state below for easy reference. Also note that, by Corollary 7.17, W is an irreducible variety for every pattern π and hence the lemma is applicable. □

Lemma 7.22 (Lemma 1, Page 124 [Sha94b]). *If X is an irreducible algebraic variety and $Y \subsetneq X$ a proper subvariety then the set $X(\mathbb{C}) - Y(\mathbb{C})$ is dense in $X(\mathbb{C})$.*

7.6 Conclusions and Open Questions

In this work, we considered two questions regarding matrix rigidity, namely constructing an explicit family of matrices that are rigid and the semicontinuity question about rigidity. The implication of the work in the computational complexity setting can be stated (in a simplified version) as follows:

Theorem 7.23 (Superlinear Lowerbound). *Let $\delta(n) = n^{4n^4}$. Let $p_{i,j}$ for $1 \leq i,\ j \leq n$ be distinct primes such that $p_{i,j} > \delta(n)$. Let $A(n) := (\zeta_{i,j}) \in \mathbb{C}^{n\times n}$. $\zeta_{i,j} = e^{2\pi \mathrm{i}/p_{i,j}}$. Then, any linear circuit over $\mathbb{C}$ of depth $O(\log n)$ computing $x \to Ax$ must have size $\Omega(n)$.*

Let $B(n) := (\zeta_{i,j} + \overline{\zeta_{i,j}}) \in \mathbb{R}^{n\times n}$. $\zeta_{i,j} = e^{2\pi \mathrm{i}/p_{i,j}}$. Then, any linear circuit over $\mathbb{R}$ of depth $O(\log n)$ computing $x \to Ax$ must have size $\Omega(n)$.

As we stated in the introduction, the above family of matrices is not explicit as in the sense of definition 6.20. To this end, we can hope to use using rational approximations of small size to the entries of the matrix in order to obtain a rigid matrix with rational entries. This motivated the study of semiconituity. Although we could prove that there are explicit matrices over complex numbers where the semi-continuity property holds, the bounds we have for the degree and coefficients on the polynomial are too weak in order to provide a polynomial size approximation for the entries which preserves the rigidity. Hence improving the bounds is also an interesting direction for further research.

Another important line of research is extending this approach to prove lower bounds for other more explicit families of matrices. As one can easily see, the potential of the approach in this work is far from fully exploited. In particular, it will be interesting to prove properties about the elimination ideal which are useful in deriving optimal lower bounds for matrix rigidity for other families of matrices. Moreover, in section 7.4, we could argue that it is sufficient to prove properties about the elimination ideals that arises out of determinantal ideals. Again, since determinantal ideals are more special (for example, they are prime ideals), it is conceivable that the elimination ideals have more properties.

In an attempt to combine our approach to other known approaches, it might be interesting to see how it compares with that of Lokam [Lok06]. Notice that both techniques penetrate the combinatorial barrier using seemingly different techniques. In addition, it seems that there is a close connection between the notion of algebraic dimension used in [Lok06] and the well studied mathematical notion of Hilbert functions. We think that a general theory of lower bounds based on this approach will unify the two directions and will be interesting in this context.

Chapter 8

Complexity of Computing Matrix Rigidity

In chapter 5 we discussed the complexity of computing the rank of a matrix exactly, and tried to characterise small circuit complexity classes based on restricted versions of the problem. However, corresponding optimisation search problems can be considerably harder. We saw in chapter 6 different optimisation versions associated with rank of a matrix.

In this chapter, we study the complexity of computing matrix rigidity. In particular, we examine the complexity of the following questions and obtain completeness results for small (counting logspace or smaller) classes: (a) determining whether $k \in O(1)$ changes to a matrix suffice to bring its rank below a specified value, and (b) constructing a singular matrix *closest* (in a restricted sense) to the given matrix. We then consider bounded rigidity, where the magnitude of individual changes is bounded by a pre-specified value, and show NP hardness in general, and tighter bounds in special cases. Most of the results in this chapter appear in [4].

8.1 Introduction

Over any field, computing rank is known to be in NC [Mul87]. Now consider the following existential search question corresponding to matrix rigidity: Given a matrix M over a ring $\mathbb{K}$, a target rank r and a bound k, decide whether the rank of M can be brought down to below r by changing at most k entries of M.

As indicated in chapter 6, the main motivation for studying rigidity is that good lower bounds on rigidity give important complexity-theoretic results in other computational models, like linear algebraic circuits and communication complexity. Though the question we address is in fact a computational version of rigidity, it has no direct implications for these lower bounds. However, it provides natural complete problems based on linear algebra for important complexity classes.

An important aspect of computing rigidity is its possible connection to the theory of natural proofs developed by Razborov and Rudich [RR97]. Valiant's reduction [Val77] (stated in Theorem 6.14) identifies "high rigidity" as a combinatorial property of functions, based on which he proves linear-size lower bounds for $\log$-depth circuits. However, we do not know how the theory of natural proofs applies to the model of arithmetic circuits. This has not been studied in sufficient detail in order to draw conclusions about the power of the proof technique. Nevertheless, this could be thought of as motivation for the computational question of rigidity.

Our question bears close resemblance to the body of problems considered under *matrix completion*, see for instance [BFS97, Lau01]. Given a matrix with indeterminates in some locations, can we instantiate them in such a way that some desired property (e.g. non-singularity) is achieved? In section 8.2.2, we discuss how results from matrix completion can yield upper bounds for our question.

Since even an upper bound of NP is not obvious, we would like to restrict the choice available in changing matrix entries. We consider two variants:

1. In the input, a finite subset $S \subseteq \mathbb{K}$ is given. M has entries over S, and the changed entries must also be from S; rank computation continues to be over $\mathbb{K}$. (For instance, we may consider Boolean matrices, so $S = \{0, 1\}$, while rank computation is over $\mathbb{K}$.) It is easy to see that this variant is indeed in NP , and in NC if $\mathbb{K}$ is a field and $k \in O(1)$.

2. In the input, a bound θ is given. We require that the changes be bounded by θ; we may apply the bound to each change, or to the total change, or to the total change per row/column. (Recall definitions 6.8) and 6.9). This version has close connections with another well-studied area called linear interval equations which arises naturally in the context of control systems theory (see [Roh96]).

We obtain tighter lower and upper bounds for some of these questions. We obtain a completeness result of $\mathsf{C_{=}L}$ when $k \in O(1)$ in the first variant, of NP when $r = n - 1$ in the second variant, and of $\mathsf{C_=L}$ when $r = n$ in the general case. The table below summaries the results.

$\mathbb{K}$, $S \subset \mathbb{K}$ (if *, then $S = \mathbb{K}$)	restriction	bound
$\mathbb{F}_p$, $*$		in NP
$\mathbb{F}_p$, $*$	$k \in O(1)$	$\mathsf{Mod}_p\mathsf{L}$-complete
$\mathbb{Z}$ or $\mathbb{Q}$, {0,1} or specified in input		in NP
$\mathbb{Z}$ or $\mathbb{Q}$, {0,1}	$k \in O(1)$	$\mathsf{C_=L}$-complete
$\mathbb{Z}$ or $\mathbb{Q}$, $*$	$k \in O(1)$	$\mathsf{C_=L}$-hard
$\mathbb{F}_p$, $*$	$r = n - 1$	$\mathsf{Mod}_p\mathsf{L}$-complete witness-search in $\mathsf{Mod}_p\mathsf{L}$
$\mathbb{Q}$, $*$	$r = n - 1$	$\mathsf{C_=L}$-complete witness-search in $\mathsf{L}^{\mathsf{GapL}}$
$\mathbb{Z}$, $*$	$r = n - 1$ and $k = 1$	in $\mathsf{L}^{\mathsf{GapL}}$
$\mathbb{Z}$ or $\mathbb{Q}$, $*$	bounded rigidity	NP-hard
$\mathbb{Z}$ or $\mathbb{Q}$, $*$	bounded rigidity, $r = n - 1$	NP-complete
$\mathbb{Z}$ or $\mathbb{Q}$, $*$	bounded rigidity, $r = n - 1, k = 1$	In PL, and $\mathsf{C_=L}$-hard

Table 8.1: Bounds on RIGID when $k \in O(1)$ or $r = n - 1$

8.2 Basic complexity results in rigidity

We recall the necessary definitions. In particular the rigidity function, and its decision version, are as defined below (Here support$(N) = \#\{(i,j) \, | N(i,j) \neq 0\}$.)

$$R_M(r) \overset{def}{=} \inf_N \; \{\text{support}(N) : rank(M+N) < r\}$$
$$\text{RIGID}_{\mathbb{K}} = \{(M, r, k) \mid R_M(r) \leq k\}$$

8.2.1 Some Basic Approaches

Intuitively, one would expect $\text{RIGID}_{\mathbb{K}}$ to be in $\exists \cdot \mathsf{NC}$: guess k locations where M is to be changed, guess the new entries to be inserted there, and compute the rank. However, this intuition, while correct for finite fields, does not directly translate to a proof for infinite fields, since the required new entries may not have representations polynomially-bounded in the input size.

We formulate the problem in terms of polynomials. Given M, r, k, guess the k positions where the entries need to be changed, and assign distinct variables $(x_1, \ldots, x_k)$ to each of those positions. Now, to achieve rank r we just need to find a common zero of $t = \binom{n}{r+1}$ polynomials which are the $(r+1) \times (r+1)$ minors of the matrix. Let $f_1, \ldots, f_t$ be these polynomials, and let I be the ideal generated by them over $\mathbb{K}[x_1, \ldots x_k]$. This is equivalent to checking if the algebraic variety defined by the t minors is non-empty.

When $\mathbb{K}$ is algebraically closed (say when $\mathbb{K} = \mathbb{C}$ or $\mathbb{R}$) we can use the tools from algebraic geometry to check this. A basic tool is the weak form of Hilbert's Nullstellensatz (Ch. 4, Thm. 1 in [CLO07]) which states as follows:

Proposition 8.1 (Weak Hilbert Nullstellensatz)**.** *Let I be an ideal in $\mathbb{K}[x_1, \ldots, x_k]$. Then $V(I) = \phi$ if and only if $I = \mathbb{K}[x_1, \ldots x_k]$.*

Now we use the notion of Gröbner basis (see Appendix B). Let G_I be the reduced Gröbner basis of the ideal I w.r.t the standard term ordering. From the Hilbert Nullstellensatz, and the definition (B.6) of reduced Gröbner basis, it follows that $V(I) = \phi$ if and only if $G_I = \{1\}$. Thus,

Proposition 8.2. *Given polynomials $f_1, \ldots, f_t$, there are no solutions to the system $f_1 = 0, f_2 = 0, \ldots, f_t = 0$ in $\mathbb{K}$ if and only if $G_I = \{1\}$.*

Thus we just need to compute the reduced Gröbner basis of the ideal generated by the f_i's. Using the algorithm due to Buchberger [Buc83] this gives a double exponential time upper bound. Since there could be exponentially many possible minors this gives the following:

Theorem 8.3. *When $\mathbb{K}$ is algebraically closed,* $\text{RIGID}_{\mathbb{K}}$ *can be solved in non-deterministic triple exponential time.*

As one can see, this is far from satisfactory. Before we end this subsection, we remark that if the underlying field is $\mathbb{R}$, there is another natural approach. Consider, $f = \sum_{i=1} f_i^2$. Clearly, the zeroes of f will be common zeroes of the f_is. Thus, finding the roots of multivariate polynomial f in $\mathbb{R}$ will be enough to test if $R_M(r) \leq k$. However, this does not seem to give a better bound in general. In the following, using a connection to matrix completion problems, we derive PSPACE upper bound for the problem over both $\mathbb{C}$ and $\mathbb{R}$.

8.2.2 Connection with Matrix Completion Problems

Now we discuss the connections between matrix rigidity and matrix completion problems. A *mixed matrix* is a matrix in which every entry is either a number or an indeterminate. Matrix completion problems, in general, ask if there is a choice of entries for the indeterminates (this is called a *completion*) such that the resulting matrix M satisfies a property P.

More formally, Let $\mathbb{F}$ be a field, and $x_1, \ldots, x_k$ be variables. Let $\mathbb{K} = \mathbb{F}[x_1, \ldots, x_k]$. A matrix $M \in (\mathbb{F} \cup \{x_1, \ldots x_k\})^{n \times n}$ is called a *mixed matrix*. In our context we consider the property P as : $\operatorname{rank}(M) < r$ for a given r. Such instances are called *minrank completion problems*, see for instance [BFS97].

Definition 8.4 (Minrank Completion). *Let $\mathbb{F}$ be a field. Given a matrix M with entries from $\mathbb{F} \cup \{x_1 \ldots x_k\}$,*

$$\textsc{MinRank}_{\mathbb{F}}(M) = \min_{(\alpha_1, \ldots \alpha_k) \in \mathbb{F}^k} \operatorname{rank}_{\mathbb{F}}(M(\alpha_1, \ldots \alpha_k))$$

Given M and r, decide if $\textsc{MinRank}_{\mathbb{F}}(M) \leq r$.

1-MinRank is a special case of the above problem where each x_i occurs in M at most once. We can easily reduce Rigid to 1-MinRank. An NP machine can simply guess the entries to be changed, and put variables in those positions, and check if there exists a choice of values for those positions such that rank goes below r. Thus we have :

Proposition 8.5. Rigid $\in$ NP(1-MinRank). *In particular,*

$$\text{1-MinRank} \in \mathsf{NP} \implies \text{Rigid} \in \mathsf{NP}$$

However, the best known upper bound for 1-MinRank over $\mathbb{Q}$ is recursively enumerability. A major open problem here, thus, is to obtain a better upper bound (e.g., decidability is not known) for the computational rigidity question.

But the approach is useful to give a weak upper bound in the following setting. For MinRank, Buss et.al.[BFS97] proves PSPACE upper bounds for over real($\mathbb{R}$) and complex($\mathbb{C}$) numbers and decidability over the p-adic numbers ($\mathbb{Q}_p$). Using these we get the following proposition,

Theorem 8.6. *If the input matrix is over $\mathbb{Q}$,* $\textsc{Rigid}_{\mathbb{C}}$*, and* $\textsc{Rigid}_{\mathbb{R}}$ *are decidable in* PSPACE*, and* $\textsc{Rigid}_{\mathbb{Q}_p}$ *is decidable.*

However, in the case of arbitrary infinite fields, the best upper bound we can see in the general case is recursive enumerability, and in particular, this is the situation over $\mathbb{Q}$. We also do not know any lower bounds for this question over $\mathbb{Q}$. In the rest of this chapter, we explore the computational complexity of several variants of this problem.

8.2.3 Characterisations when k is a constant

We now study the decision version of rigidity $\textsc{Rigid}_{\mathbb{K}}$, and also its restriction $\textsc{Rigid}_{\mathbb{K},S}$ defined below, where the matrices can have entries only from $S \subseteq \mathbb{K}$.

$$\textsc{Rigid}_{\mathbb{K},S} = \left\{ (M,r,k) \mid \begin{array}{c} M \text{ over } S,\ \ \exists M' \text{ over S} : \\ \text{rank}(M') < r\ \wedge\ \text{support}(M - M') \leq k \end{array} \right\}$$

We will mostly consider S to be either all of $\mathbb{K}$, or only We also consider the complexity of $\textsc{Rigid}$ when k is fixed, via the following language:

$$\textsc{Rigid}_{\mathbb{K},S}(k) = \{(M,r) \mid (M,r,k) \in \textsc{Rigid}_{\mathbb{K},S}\}$$

The language $\textsc{Rigid}_{\mathbb{Z}}(0)$ is nothing but $\textsc{rank bound}(\mathbb{Z})$ (see definition in Section 5.1) , and hence by [ABO96] is complete for $\mathsf{C_{=}L}$. When $k > 0$, we can still obtain some bounds provided S is finite. We have the following completeness result for one such case.

Theorem 8.7. *For each k, $\textsc{Rigid}_{\mathbb{Z},\mathbb{B}}(k)$ is complete for $\mathsf{C_{=}L}$.*

Proof. **Membership:** We show that for each k, $\textsc{Rigid}_{\mathbb{Z},\mathbb{B}}(k)$ is in $\mathsf{C_{=}L}$. An instance (M,r) is in $\textsc{Rigid}_{\mathbb{Z},\mathbb{B}}(k)$ if there is a set of $0 \leq s \leq k$ entries of M, which, when flipped, yield a matrix of rank below r. The number of such sets is bounded by $\Sigma_{s=0}^{k}\binom{n^2}{s} = t \in n^{O(1)}$. Let the corresponding matrices be denoted $M_1, M_2 \ldots M_t$; these can be generated from M in logspace. Now $(M,r) \in \textsc{Rigid}_{\mathbb{Z},\mathbb{B}}(k) \iff \exists i : (M_i, r) \in \textsc{rank bound}(\mathbb{Z})$. Hence $\textsc{Rigid}_{\mathbb{Z},\mathbb{B}}(k) \leq_{dtt}^{\log} \textsc{rank bound}(\mathbb{Z})$. Since $\textsc{rank bound}(\mathbb{Z})$ is in $\mathsf{C_{=}L}$, and since $\mathsf{C_{=}L}$ is closed under logspace disjunctive truth-table reductions [AO96], it follows that $\textsc{Rigid}_{\mathbb{Z},\mathbb{B}}(k)$ is in $\mathsf{C_{=}L}$.

Hardness: Now we show a corresponding hardness result: The hardness for $\textsc{Rigid}_{\mathbb{Z},\mathbb{B}}(0)$ is easy to see. Indeed, M is singular if and only if $(M, n-1) \in \textsc{Rigid}_{\mathbb{Z},\mathbb{B}}(0)$. In addition, $\textsc{singular}$ remains $\mathsf{C_{=}L}$-hard even when restricted to 0-1 matrices. Hardness for other values of k follows from this fact, and from the following claim. Let $N_k = M \otimes I_{k+1}$, where

$\otimes$ denotes tensor product. Note that $\mathrm{rank}(N_k) = (k+1)\,\mathrm{rank}(M)$. Now,

$$(1)\ M \in \textsc{singular}(\mathbb{Z}) \implies (N_k, (n-1)(k+1)-k) \in \textsc{Rigid}_{\mathbb{Z},\mathbb{B}}(0) \subseteq \textsc{Rigid}_{\mathbb{Z},\mathbb{B}}(k)$$
$$(2)\ M \notin \textsc{singular}(\mathbb{Z}) \implies (N_k, (n-1)(k+1)-k) \notin \textsc{Rigid}_{\mathbb{Z}}(k)$$

To see this, observe that if $M \in \textsc{singular}(\mathbb{Z})$, then $\mathrm{rank}(M) \leq n-1$ and so $\mathrm{rank}(N_k) \leq (k+1)(n-1) < (n-1)(k+1)-k$. Thus $R_{N_k}((n-1)(k+1)-k) = 0$. Thus $(N_k, (n-1)(k+1)-k) \in \textsc{Rigid}_{\mathbb{Z},\mathbb{B}}(0)$. If $M \notin \textsc{singular}(\mathbb{Z})$, then $\mathrm{rank}(N_k) = (n-1)(k+1)$, and by Lemma 6.5, $R_{N_k}(n(k+1)-k) > k$. Thus, $(N_k, n(k+1)-k) \notin \textsc{Rigid}_{\mathbb{Z}}(k)$. □

However, in the case when the underlying ring is infinite, where S is given more implicitly with size not exponentially bounded, the changed entry might not have a polynomial sized representation in terms of the input size, this approach fails.

Now we look for analogues of these results over finite fields of the form $\mathbb{F}_p$ where p is prime. We first pinpoint the complexity of computing rank. It is known [BDHM92] that $\textsc{singular}(\mathbb{F}_p)$ is complete for $\mathsf{Mod}_p\mathsf{L}$. (In fact, computing the exact value of the determinant over $\mathbb{F}_p$ is in $\mathsf{Mod}_p\mathsf{L}$.) We observe that so is $\textsc{rank bound}(\mathbb{F}_p)$.

Lemma 8.8. *For prime p, $\textsc{rank bound}(\mathbb{F}_p)$ is in $\mathsf{Mod}_p\mathsf{L}$.*

Proof. Over an arbitrary finite field, Mulmuley's algorithm [Mul87] reduces the problem of computing the rank of a matrix to testing whether certain coefficients of the characteristic polynomial of a related (univariate) polynomial matrix are all 0. Over $\mathbb{Z}$, each coefficient of the characteristic polynomial can be computed in GapL ; hence checking that it is 0 in the field $\mathbb{F}_p$ can be tested in $\mathsf{Mod}_p\mathsf{L}$. Since $\mathsf{Mod}_p\mathsf{L}$ is closed under conjunctive truth-table reductions [AO96], it follows that $\textsc{rank bound}(\mathbb{F}_p)$ is also in $\mathsf{Mod}_p\mathsf{L}$. □

Now we can obtain analogues of Theorem 8.7 using Lemma 8.8 and the fact that $\mathsf{Mod}_p\mathsf{L}$ is closed under disjunctive truth-table reductions [AO96].

Theorem 8.9. *For each k, and each prime p, $\textsc{Rigid}_{\mathbb{F}_p}(k)$ is complete for $\mathsf{Mod}_p\mathsf{L}$.*

The hardness results above were obtained essentially by exploiting the hardness of testing singularity. Therefore we now consider the complexity of $\textsc{Rigid}$ at the singular-vs-non-singular threshold, i.e. when $r = n$.

From Proposition 6.5, we know that over any field $\mathbb{F}$, $(M, n-1, k)$ is in $\textsc{Rigid}$ whenever $k \geq 1$. And $(M, n-1, 0)$ is in $\textsc{Rigid}$ if and only if $M \in \textsc{singular}(\mathbb{F})$. So the complexity of deciding this predicate over $\mathbb{Q}$ is already well understood. We then address the question of how difficult it is to come up with a witnessing matrix.

Theorem 8.10. *Given a non-singular matrix M over $\mathbb{Q}$, a singular matrix N satisfying support$(M - N) = 1$ can be constructed in L^{GapL}.*

Proof. For each (i, j), let $M(i, j)$ be the matrix obtained from M by replacing $m_{i,j}$ with an indeterminate x. Then det$(M(i, j))$ is of the form $ax + b$, and a and b can be determined in GapL (see for instance [AAM03]). Since $R_M(n-1) = 1$, there is at least one position (i, j) where the determinant is sensitive to the entry, and hence $a \neq 0$. Setting $m_{i,j}$ to be $-b/a$ gives the desired N. □

Another question that arises naturally is the complexity of RIGID at the singularity threshold over rings. Note that Proposition 6.5 does not necessarily hold for rings. For instance, changing one entry of a non-singular rational matrix M suffices to make it singular. But even if M is integral, the changed matrix may not be integral, and over $\mathbb{Z}$, $R_M(n-1)$ may well exceed 1. (It does, for the matrix $\begin{bmatrix} 2 & 3 \\ 5 & 7 \end{bmatrix}$.) Thus, the question of deciding $R_M(n)$ over $\mathbb{Z}$ is non-trivial. We show:

Theorem 8.11. *Given $M \in \mathbb{Z}^{n\times n}$, deciding if $(M, n-1, k)$ is in* RIGID*($\mathbb{Z}$) is (1) trivial for $k \geq n$, (2)* $\mathsf{C_=L}$ *complete for $k = 0$, and (3) in* $\mathsf{L}^{\mathsf{GapL}}$ *for $k = 1$.*

Proof. The first part holds because zeroing out an entire row always gets singularity. The second part merely says that SINGULAR($\mathbb{Z}$) is $\mathsf{C_=L}$-complete. For the third part, we use the idea from the proof of Theorem 8.10. $(M, n, k) \in$ RIGID($\mathbb{Z}$) if and only if there exists a position (i, j) such that det$(M(i, j)) = ax + b$ and b/a is integral. A logspace machine can check integrality of b/a, obtaining the bits of a and b by querying GapL. □

In particular, the third result in this theorem implies that if over $\mathbb{Z}$, $R_M(n) = 1$ for a non-singular matrix M, and if N is the witnessing matrix, then the single non-zero entry in N has size polynomially bounded in that of M. However, if $R_M(n) > 1$ we do not know such a size bound.

8.2.4 Inapproximability results on Rigidity

Now we observe that inapproximability results for rigidity over $\mathbb{F}_2$ follow from a reduction from the nearest codeword problem (NCP) discovered by Deshpande [Des07]. We need some background from coding theory:

A linear code is a linear subspace of the vector space $\mathbb{F}^n$ along with an encoding function $\mathcal{E} : \mathbb{F}^m \to \mathbb{F}^n$ and a decoding function $\mathcal{D} : \mathbb{F}^n \to \mathbb{F}^m$. The *generator matrix* $G \in \mathbb{F}^{m\times n}$

provides the encoder as a linear transformation from $\mathbb{F}^m \to \mathbb{F}^n$. Given $x \in \mathbb{F}^m$, the codeword corresponding to x is given by xG, and the set of codewords is defined by $\mathcal{C} = \{xG \in \mathbb{F}^n \mid x \in \mathbb{F}^m\}$. Equivalently, there exists a parity check matrix $H \in \mathbb{F}^{(n-m)\times n}$ such that $\mathcal{C} = \{x \in \mathbb{F}^m \mid Hx = 0 \in \mathbb{F}^{n-m}\}$.

Definition 8.12 (Nearest Codeword Problem - NCP(H, z, d)). *Given the parity check matrix of a linear code $H_{(n-m)\times n}$, a received vector $z \in \mathbb{F}^n$, and distance d, check if there a vector $y \in \mathbb{F}^n$ such that $Hy = 0$ and $\Delta(z, y) \leq d$.*

Reduction from NCP to Rigidity [Des07]: Let G and H be the generator matrix and parity check matrix of the linear code respectively. Construct an $(m(n+1)+1) \times (n)$ matrix M by putting $z = 1^n$ in the first row and stacking $(n+1)$ copies of G below it. We assume that the generator matrix is full row-rank. Then the rank of M is m if z is in the row span of G, and $m+1$ otherwise. Now the rigidity instance that we produce is (M, m, d). Thus the factors appear in terms of the number of columns of the given matrix. The following lemma establishes the correctness of the reduction.

Lemma 8.13 ([Des07]). $R_M(m) \leq d \iff (H, z, d) \in$ NCP.

We get the inapproximability results by directly combining the above Lemma with the known inapproximability results for NCP [ABSS93].

Theorem 8.14. *Over $\mathbb{F}_2$, for any constant $\alpha > 1$, given a matrix $M \in \mathbb{F}_2^{m\times n}$ of rank r, deciding if $R_M(r-1) \leq k$ or $R_M(r-1) \geq \alpha k$ is* NP*-hard.*

Theorem 8.15. *Assuming* NP *is not contained in* DTIME$(n^{\log n})$*, over $\mathbb{F}_2$, for any $\epsilon > 0$, for $\alpha \leq 2^{n \log^{0.5-\epsilon} n}$, given a matrix $M \in \mathbb{F}_2^{m\times n}$, of rank r it is impossible to distinguish between the following two cases:*

1. $R_M(r-1) \leq k$.
2. $R_M(r-1) \geq \alpha k$.

8.2.5 A Maximisation Version of Rigidity

In this section, we consider a variant of the problem of matrix rigidity where we ask for the minimum number of entries needed to bring up the rank above a given value r. We start with the following definitions.

$$\overline{R}_M(r) \overset{def}{=} \inf_N \ \{\text{support}(N) : rank(M+N) \geq r\}$$

$$\text{MAXRIGID} = \{(M,r,k) \mid \overline{R}_M(r) \leq k\}$$

Intuitively, one would expect that MAXRIGID is easier because to ensure that the rank of the matrix is at least r, is it sufficient to change values such that at least one $r \times r$ minor becomes non-zero. We show that much more is true.

Lemma 8.16. $\overline{R}_M(r) = \max\{0, r - \text{rank}(M)\}$.

Proof. If $r \leq \text{rank}(M)$ then there is nothing to prove. So we assume that $rank(M) < r$. Using Fact 6.6, it is easy to see that $\overline{R}_M(r) \geq r - \text{rank}(M)$. Indeed, a change in the matrix can change the rank by at most 1.

To see the other direction, let the rank of the matrix be ℓ. If $\ell \geq r$ we have nothing to prove. So assume that $\ell < r$. Choose an $\ell \times \ell$ submatrix M_0 of M which has non-zero determinant. Without loss of generality, we can assume that M_0 is the top left corner of M. Replace $(\ell+1, \ell+1)^{\text{th}}$ entry with a variable x. Let M_1 be the top-left $(\ell+1) \times (\ell+1)$ submatrix of M. The determinant of M_1 is of the form $ax+b$ where $a = det(M_0)$. Choose $x \neq -(b/a)$ thus ensuring $det(M_1)$ is non-zero. Repeat this procedure choosing M_1 to be the new M_0 and $\ell+1$ to be the new ℓ. At each step we are increasing the rank of M by 1, by changing exactly one entry, and hence by $r - rank(M)$ steps we will reach a matrix M' of rank at least r which differs from M in at most $r - rank(M)$ positions. This $\overline{R}_M(r) \leq r - \text{rank}(M)$. □

Thus, to test if $(M,r,k) \in$ MAXRIGID it is sufficient to test if $\text{rank}(M) > r-k-1$. Hence, by proposition 5.2, we have the following theorem.

Theorem 8.17. *Over $\mathbb{Z}$ or $\mathbb{Q}$,* MAXRIGID *is* coC$_=$L*-complete. Over $\mathbb{F}_p$,* MAXRIGID *is* Mod$_p$L*-complete*

In the above proof, in order to choose x at each stage, we need to compute the coefficients a and b which are determinants themselves. This gives a polynomial time procedure which given M and r computes the actual changes that we need to make to bring up the rank of the matrix to r or more.

We remark that this characterisation (Lemma 8.16) and the upper bound for max version of the rigidity, brings out a subtle difference between the rigidity-like problems and matrix completion problems considered in the literature. To make this clearer, we describe the maximum rank matrix completion problem (the max version of Section 8.2.2, studied in [BFS97]).

Given a matrix with indeterminates and a value r the problem is to check if there is a choice of values for the indeterminates such that the matrix (when the indeterminates are substituted with these values) has rank r or more. A special case of this problem (called 1-MAXRANK) is when each indeterminate can appear only once in the matrix. An easy argument similar to that of Proposition 8.5 gives: MAXRIGID $\in \mathsf{NP}\,(\text{1-MAXRANK})$. Notably, 1-MAXRANK also has a polynomial time upper bound. Using more involved arguments from matroid theory Murota [Mur93] and Geelen [Gee99] independently gave polynomial time algorithms for the problem.

8.3 Computing Bounded Rigidity

We now consider the bounded norm variant of rigidity which was described in Chapter 6. Namely, changed matrix entries can differ from the original entries by at most a pre-specified amount θ. Note that over $\mathbb{Q}$, this still does not imply an a priori polynomial-size bound on the changed entries.

Recall the definition of *norm rigidity* $\Delta_M(r)$ and *bounded rigidity* $R_M(r, \theta)$ from chapter 6. We define a corresponding decision version:

$$\text{B-RIGID}_{\mathbb{K}} = \{(M, r, k, \theta) \mid R_M(r, \theta) \leq k\}.$$

Recall from Lemma 6.10 that there are cases when $R_M(r, \theta)$ is undefined. This motivates the following question. Given a matrix M, a rank r and θ how difficult is it to check whether $R_M(r, \theta)$ is defined? We show the following:

Theorem 8.18. *1. Given a matrix $M \in \mathbb{Q}^{n\times n}$, and a rational number $\theta > 0$, testing if $R_M(n-1, \theta)$ is defined is* NP*-complete.*

2. Given M and θ as above, and further given an integer k, testing if $R_M(n-1, \theta)$ is at most k is NP*-complete.*

Proof. To begin with, notice that, $R_M(n-1, \theta)$ is defined if and only if $R_M(n-1, \theta) \leq n^2$.
Membership: We first show the membership in NP for (2). Membership in (1) follows by using this with $k = n^2$. We use notation and some results from the *linear interval equations*

literature. For two matrices A and B, we say that $A \leq B$ if for each i,j, $A_{ij} \leq B_{ij}$. For $A \leq B$, the interval of matrices $[A, B]$ is the set of all matrices C such that $A \leq C \leq B$. An interval is said to be singular if it contains at least one singular matrix; otherwise it is said to be regular. By Theorem 2.8 of [PR93] (or directly from Lemma 8.20), checking singularity of a given interval matrix is in NP.

Given M, θ and k, we want to test whether $R_M(n, \theta)$ is at most k. In NP, we guess k positions $(i_1, j_1), (i_2, j_2), \dots (i_k, j_k)$ and construct the matrix $V_{i_m j_m} = \theta$ for all $1 \leq m \leq k$ and 0 elsewhere. Now let $\underline{A} = M - V$ and $\overline{A} = M + V$. Then $R_M(n-1, \theta) \leq k$ if and only if for some such guessed V, the interval $[\underline{A}, \overline{A}]$ is singular, and this can be tested in NP.

Hardness: It suffices to prove hardness for (1), since hard instances of (1) along with $k = n^2$ gives hard instances of (2). We start with the *maximum bipartite subgraph problem*: Given an undirected graph $G = (V, E)$, with n vertices and m edges and a number k, check whether there is bipartite subgraph with at least k edges. This problem is known to be NP-complete (see [GJ79]). In [PR93], there is a reduction from this problem to computing the *radius of non-singularity*, defined as follows: Given a matrix A, its radius of non-singularity $d(A)$ is the minimum $\epsilon > 0$ such that the interval $[A - \epsilon J, A + \epsilon J]$ is singular, where J is the all-1s matrix.

We sketch the reduction of [PR93] below and observe that it yields NP-hardness for our problem as well.

Given an instance G, k of the maximum bipartite subgraph problem, we define the matrix N as,

$$N_{ij} = \begin{cases} -1 & \text{if } i \neq j \text{ and } i \text{ and } j \text{ are adjacent in } G \\ 2m+1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

Notice that since N is diagonally dominant, by Levy-Desplanques theorem (see for instance Theorem 2.1 in [MM64]), N is invertible. Let $M = N^{-1}$.

By Theorems 2.6 and 2.2 of [PR93],

$$\begin{aligned} (G,k) \text{ is a Yes instance} &\iff 1/d(M) \geq (2m+1)n + 4k - 2m \\ &\iff d(M) \leq \theta = \tfrac{1}{(2m+1)n+4k-2m} \\ &\iff \text{the interval } [M - \theta J, M + \theta J] \text{ is singular} \\ &\iff R_M(n-1, \theta) \text{ is defined.} \end{aligned}$$

□

Unravelling the NP algorithm described in the membership part above, and its proof of correctness, is illuminating. Essentially, what is established in [Roh94] and used in [PR93] is the following:

Lemma 8.19 ([Roh94]). *If an interval $[A, B]$ is singular, i.e. the determinant vanishes for some matrix C within the bounds $A \leq C \leq B$, then the determinant vanishes for a matrix $D \in [A, B]$ which, at all but at most one position, takes an extreme value (d_{ij} is either a_{ij} or b_{ij}).*

In particular, this implies that there is a matrix in the interval whose entries have representations polynomially long in that of A and B. To see this, let D be the matrix claimed to exist as above, and let k, l be the (only) position where $a_{kl} < d_{kl} < b_{kl}$. The other entries of D match those of A or B and hence are polynomially bounded anyway. Now put a variable x at k, l to get matrix D_x. Its determinant is a univariate linear polynomial $\alpha x + \beta$ which vanishes at $x = d_{kl}$. Now α and β can be computed from D_x in GapL , and have polynomially bounded representations. If $\alpha = 0$, then $\beta = 0$ and the polynomial is identically zero. Otherwise, the zero of the polynomial is $-\beta/\alpha$. Either way, there is a zero with a polynomially long representation.

In [Roh94], the above lemma is established as part of a long chain of equivalences concerning determinant polynomials. However, it is in fact a general property of arbitrary multilinear polynomials, as we show below.

Lemma 8.20 (Zero-on-an-Edge Lemma). *Let $p(x_1 \ldots x_t)$ be a multilinear polynomial over $\mathbb{Q}$. If it has a zero in the hypercube H defined by $[\ell_1, u_1], \ldots [\ell_t, u_t]$, then it has a zero on an edge of H, i.e. a zero $(a_1, \ldots, a_t)$ such that for some k, $\forall (i \neq k)$, $a_i \in \{\ell_i, u_i\}$.*

Proof. The proof is by induction on the dimension of the hypercube. The case when $t = 1$ is vacuously true, since H is itself an edge. Consider the case $t = 2$. Let $p(x_1, x_2)$ be the multilinear polynomial which has a zero (z_1, z_2) in the hypercube H; $\ell_i \leq z_i \leq u_i$ for $i = 1, 2$. Assume, to the contrary, that p has no zero on any edge of H. Define the univariate polynomial $q(x_1) = p(x_1, z_2)$. Since $q(x_1)$ is linear and vanishes at z_1, $p(\ell_1, z_2)$ and $p(u_1, z_2)$ must be of opposite sign. But the univariate linear polynomials $p(\ell_1, x_2)$ and $p(u_1, x_2)$ do not change signs on the edges either, and so $p(\ell_1, u_2)$ and $p(u_1, u_2)$ also have opposite sign. By linearity of $p(x_1, u_2)$, there must be a zero on the edge $x_2 = u_2$, contradicting our assumption.

Let us assume the statement for hypercubes of dimension less than t. Consider the hypercube of dimension t and the polynomial $p(x_1, \ldots x_t)$. Let $(z_1 \ldots z_t)$ be the zero inside the hypercube. The multilinear polynomial r corresponding to $p(x_1, \ldots x_{n-1}, z_t)$ has a zero inside the $(t-1)$-dimensional hypercube H' defined by intervals $[\ell_1, u_1], \ldots [\ell_{t-1}, u_{t-1}]$. By induction, r has a zero on an edge of H'. Without loss of generality, assume that this zero is $(z_1', \alpha_2 \ldots \alpha_{t-1})$ where $\alpha_i \in \{\ell_i, u_i\}$. Thus the polynomial $q(x_1, x_t) = p(x_1, \alpha_2 \ldots \alpha_{t-1}, x_t)$ has a zero in the hypercube defined by intervals $[\ell_1, u_1], [\ell_t, u_t]$. Hence the base case applies again, completing the induction. □

The hard instance that we get in Theorem 8.18 is a matrix with a rational entries and the bound θ is also a rational number. If M is such a matrix, we can produce an integer matrix N with the same rank by multiplying each entry by ℓ where ℓ is the lcm of the denominators of the entries. $R_M(r, \theta) = R_N(r, \ell\theta)$. Thus, theorem 8.18 hold for integer matrices too, and recall that the rank of an integer matrix over $\mathbb{Z}$ and $\mathbb{Q}$ are the same.

Remark 8.21. *The matrices that are produced in the above reduction are all symmetric. Rohn [Roh94] considered the case when the interval of matrices under consideration is symmetric; that is both the boundary matrices are symmetric. Notice that the interval can still contain non-symmetric matrices. He proved that in such an interval, if there is a singular matrix, then there must be a symmetric singular matrix too. So even restricted to symmetric matrices, the above result holds.*

Remark 8.22. *A set of matrices C is a convex set if for any set of matrices $M_1, \ldots, M_k \in C$ and any $\alpha_1, \ldots, \alpha_k$ such that, $\sum_i \alpha_i = 1$, the matrix $M = \sum_i \alpha_i M_i$ is also in C. Notice that in the case of an interval matrix $A = [\overline{A}, \underline{A}]$, the hypercube defined by the intervals is a convex set, where the corners of the set(polytope) are the 2^{n^2} matrices :*

$$\left\{ \tilde{A} \mid \tilde{A}_{ij} = \overline{A}_{ij} \text{ or } \underline{A}_{ij} \right\}$$

Thus, the above reduction also implies the hardness of optimising rank over a convex set C (of m corners) of matrices when the set is represented by encoding its corners using $\log m$ bits.

Analogous to Theorem 8.7, we consider the complexity of B-RIGID$_{\mathbb{K}}$ when $k \in O(1)$.

Theorem 8.23. B-RIGID$_{\mathbb{K}}$ *is* C$_=$L*-hard for each fixed choice of k, and remains hard when r is restricted to be $n-1$. When $k = 1$ and $r = n-1$, it is in* PL.

Proof. For any k, $(M, n, k, 0) \in$ B-RIGID$_{\mathbb{K}}$ $\iff$ M is singular; hence C$_=$L-hardness.

To see the PL upper bound, let $\theta = \frac{p}{q}$. For each element (i, j), define the the $(i, j)^{th}$ element as variable x and then write the determinant as $ax + b$. Thus, if $|x| = |\frac{b}{a}| \le \frac{p}{q}$ for at least one such (i, j) pair, we are done. This is equivalent to checking if $(bq)^2 \le (ap)^2$. a and b can be written as determinants, hence $(ap)^2$ and $(bq)^2$ are GapL functions, and comparison of two GapL functions can be done in PL. Since PL is closed under disjunction (see [AO96]), the entire computation can be done in PL. □

Appendix A

Complexity Theory Preliminaries

In this chapter we recall the basic complexity theory that is needed this thesis. We refer the reader to any standard textbook on complexity theory ([DK00]), and to a survey article by Fortnow and Homer [FH03] for a historical perspective and pointers to many important results of complexity theory.

The following notation is from classical automata theory : an *alphabet* is a finite set Σ (in our case $\{0, 1\}$)) of *symbols*. A finite sequence $x = x_1 \ldots x_n$ where each $x_i \in \Sigma$ forms a *string* of *length* n, and we will use |x| to denote the length of x. Σ^* denotes the set of all strings over Σ. A *language* L over Σ is a subset of Σ^*. Computational problems are often posed as *decision problems* where the answer that we expect is either *yes/no*. By suitably encoding with binary strings they define a language L over $\{0, 1\}^*$. In contexts where we expect more than one bit as output, the problem is thought of as a function from $\{0, 1\}^* \to \{0, 1\}^*$. They will be called *functional problems*.

Definition A.1. *The complexity class* P *is the class of decision problems that can be solved by a Turing machine in time bounded by a polynomial in the size of the input.*

The robustness properties of this class motivated Edmond [Edm65] to suggest, what is now widely accepted as the *Extended Church-Turing thesis,* that P is the class of problems which are *tractable* under any reasonable model of computation.

To abstract out the properties of decision problems that could be outside this class the notion of non-determinism was introduced. In this model, the machine is allowed to guess a solution and verify if it is indeed one, and is said to accept if at least one of them is indeed a solution. This captures efficient verifiability property of a candidate solution that appears to be the structure of certain languages that are seemingly not efficiently computable.

Definition A.2. *The class of decision problems, that can be solved by non-deterministic polynomial time bounded Turing machines is called* NP.

It is a big open question [Coo03] in complexity theory if there is a language in NP which is provably not in P.

In an attempt to understand the structure of decision problems, classical recursion theory formalised the notion of reductions and completeness for a class of problems.

Definition A.3. *(reductions) Let A and B two languages, we say that A many-one reduces to B, denoted by $A \leq_m B$ if there is a function $f : \Sigma^* \rightarrow \Sigma^*$ that maps instances of problem A to instances of problem B such that yes and no instances of A gets mapped to yes and no instances of B respectively. Further, a reduction is said to be in* poly *time, denoted by $A \leq_m^p B$, if the function f can be computed by a polynomial time Turing machine. We say that A Turing reduces to B, denoted by $A \leq_T B$, if there is a Turing machine M for A which can ask, during its computation, some membership questions about language B. Further, if the time taken by the machine is bounded by polynomial, and the length of the query strings for membership questions asked about B, is also bounded by polynomial, then we say that A is polynomial time Turing reducible to B and denote it by $A \leq_T^p B$.*

We will also use variants of this definition based on resource bounds for the machine M.

A language L is complete for a complexity class $\mathcal{C}$ under many-one (Turing) $\mathcal{C}'$-reductions if for every language $L' \in \mathcal{C}$, L' many-one (Turing) reduces to L where the complexity of reduction is in $\mathcal{C}'$. Thus L is the *hardest* problem in $\mathcal{C}$. It is easy to see that this notion will be useful when the class $\mathcal{C}'$ is contained in $\mathcal{C}$ and is not known to be the same as $\mathcal{C}$. This gives a class of problems which are candidates for the separation of the two classes.

The notion of NP-completeness gives more candidate problems to attack for settling the P vs. NP problem. See [GJ79] for a list of NP-complete problems.

Now we turn into the space bounded complexity classes.

Definition A.4. L *denotes the class of languages accepted by deterministic Turing machines which run in space at most* $\log n$*.* NL *denotes the class of languages accepted by non-deterministic Turing machines which use at most* $\log n$ *space and for which the input available on a read-only tape.*

Reachability problems in graphs is a source of complete problems for space bounded classes in the combinatorial domain. The problem in general is of the form: given a graph G and two designated vertices s and t, determine if there is a path from s to t. A complete problem for L is the reachability problem in undirected forests [CM87]. Recent results [Rei05] show that the reachability problem in undirected graphs can also be solved in L, and hence the problem is L-complete. Reachability in directed graphs can be easily seen to be complete for NL.

Now we equip the space bounded Turing machine with a poly size bounded stack. This model called AUXPDA can accept all context free languages (since it can simulate push down automata), and in fact more.

Definition A.5. LogCFL *is the set of problems which are* $\log$ *space many one reducible to some* CFL. LogDCFL *is the set of problems which are* $\log$ *space many one reducible to some* DCFLs*.*

We now [Sud78, Coo71] (see also [Ruz80, Ven91, MRV99]) know that the class of languages accepted by the AUXPDA running in polynomial time is exactly LogCFL. Similar results are known for LogDCFLs as well.

We will also need the notion of counting in space bounded complexity classes. For a nondeterministic Turing machine M we denote by $\mathsf{acc}_M(x)$ and $\mathsf{rej}_M(x)$ the number of accepting and rejecting computation paths of M on input x respectively.

Definition A.6 (#L)**.** *The class* #L *is defined as the class of functions* $f : \Sigma^* \to N$ *for which there is a nondeterministic logspace-bounded Turing machine* M*, such that* $\forall x \in \Sigma^*$*,* $f(x) = \mathsf{acc}_M(x)$*.*

Definition A.7 (GapL)**.** *The class* GapL *is defined as the class of functions* $f : \Sigma^* \to \mathbb{Z}$ *for which there is a nondeterministic logspace-bounded Turing machine* M*,* $\forall x \in \Sigma^*$*,* $f(x) = \mathsf{acc}_M(x) - rej_M(x)$*.*

Based on these class of functions one can define the following decision problems.

Definition A.8. $\mathsf{C}_=\mathsf{L}$ *is the class of languages for which there is a function* f *in* GapL *such that*

$$\forall x \in \Sigma^* \ : \ x \in L \iff f(x) \neq 0$$

Definition A.9. PL *is the class of languages for which there is a function* f *in* GapL *such that*

$$\forall x \in \Sigma^* \ : \ x \in L \iff f(x) \geq 0$$

Definition A.10. *Let* $k \geq 2$ *be an integer.* $\mathsf{Mod}_k\mathsf{L}$ *is the class of sets* L *such that there is an* $f \in$ #L *with*

$$\forall x \in \Sigma^* \ : \ x \in L \iff f(x) \not\equiv 0 (\mathrm{mod} k)$$

We now turn to circuit complexity classes that are relevant to this thesis. We refer the reader to a standard textbook [Vol99] for a detailed exposition of this material. Circuits C over the basis $\mathcal{B}$ with n inputs $x_1, \ldots x_n$, are directed acyclic graphs (DAGs) where the nodes are labelled by gates $g \in \mathcal{B}$. The gates at the nodes with in-degree 0 are simply the input literals. The gates at the nodes with out-degree 0 are called the output nodes. The in-degree of a node is the fanin. The depth of a circuit is the length of the longest directed path in the underlying graph. The size of the circuit is simply the number of nodes in the underlying graph. Now we can define the following complexity classes. For the basis $\mathcal{B} = \{\wedge_k, \vee_k, \neg, \mod_k, maj_k\}$ denote the boolean gates for computing conjunction, disjunction, negation, modulus, and majority of fanin at most k. A circuit family $\{\mathcal{C}\}$ is said to accept a language L, if $\forall x \in \Sigma^*$, $x \in L \iff C_{|x|}$ evaluates to 1 on input x.

Definition A.11. NC^i *is the class of languages that can be computed by circuits of size* $\mathsf{poly}(n)$ *and depth* $O(\log^i n)$ *over the basis* $\{\wedge_2, \vee_2, \neg\}$. $\mathsf{NC} = \cup_i \mathsf{NC}^i$.

Definition A.12. AC^i *is the class of languages that can be computed by circuits of size* $\mathsf{poly}(n)$ *and depth* $O(\log^i n)$ *over the basis* $\{\wedge_m, \vee_m, \neg\}$ *where* $m = \mathsf{poly}(n)$. $\mathsf{AC} = \cup_i \mathsf{AC}^i$

Definition A.13. ACC^i *is the class of languages that can be computed by circuits of size* $\mathsf{poly}(n)$ *and depth* $O(\log^i n)$ *over the basis* $\{\wedge_m, \vee_m, \mod_m, \neg\}$ *where* $m = \mathsf{poly}(n)$. $\mathsf{ACC} = \cup_i \mathsf{ACC}^i$.

Definition A.14. TC^i *class of languages that can be computed by circuits of size* $\mathsf{poly}(n)$ *and depth* $O(\log^i n)$ *over the basis* $\{\wedge_m, \vee_m, maj_m, \neg\}$ *where* $m = \mathsf{poly}(n)$. $\mathsf{TC} = \cup_i \mathsf{TC}^i$.

In a spirit similar to the class GapL and $\mathsf{C_{=}L}$ one can define GapNC^1 and $\mathsf{C_{=}NC}^1$.

Definition A.15. (GapNC^1) *The class* $\#\mathsf{NC}^1$ *is defined as the class of functions* $f : \Sigma^* \to \mathbb{N}$ *for which there is a uniform family of circuits* $\mathcal{C}$ *of* $poly(n)$ *size and* $O(\log n)$ *depth such that,*

$$\forall n,\ \forall x \in \Sigma^n,\ f(x) = \mathsf{acc}_C(x)$$

where $\mathsf{acc}_C(x)$ *will denote the number of accepting subtrees of circuit* $C_n \in \mathcal{C}$ *on input* x. *Now* GapNC^1 *denotes the class of functions which can be expressed as the difference of two functions in* $\#\mathsf{NC}^1$. $\mathsf{C_{=}NC}^1$ *is a language class such that there is a function* f *in* GapNC^1*, such that* $x \in L \iff f(x) \neq 0$.

Appendix B

Algebraic Geometry Preliminaries

Here we recall some basic notions from algebraic geometry. We refer the reader to [Sha94b, Sha94a] for detailed treatment the these notions.

Let $\mathbb{F}$ be a field. Let $\overline{\mathbb{F}}$ denote the fixed algebraic closure of $\mathbb{F}$. Let $x_1, \ldots, x_n$ be n algebraically independent variables over $\mathbb{F}$. Let $\mathbb{F}[x_1, \ldots, x_n]$ be the polynomial ring in n variables over $\mathbb{F}$. An ideal I is by definition a sub-module of the ring $\mathbb{F}[x_1, \ldots, x_n]$. More explicitly, I is a subset of $\mathbb{F}[x_1, \ldots, x_n]$ which is a subgroup of $\mathbb{F}[x_1, \ldots, x_n]$ under addition, and which is also closed under multiplication by elements of $\mathbb{F}[x_1, \ldots, x_n]$.

By an *algebraic subset* $V(\Sigma)$ of $\mathbb{F}^n$, we mean the set of common zeros of a set Σ of polynomials in $\mathbb{F}[x_1, \ldots, x_n]$. If Σ is a set of homogeneous polynomials then $V(\Sigma)$ is a subspace of $\mathbb{F}^n$. If Σ is a set of non-homogeneous polynomials then $V(\Sigma)$ is either empty or translate of a subspace of $\mathbb{F}^n$.

Given a subset Σ of $\mathbb{F}[x_1, \ldots, x_n]$ we may consider the ideal $I_\Sigma = \langle \Sigma \rangle$ generated by Σ in $\mathbb{F}[x_1, \ldots x_n]$. Given an ideal I of $\mathbb{F}[x_1, \ldots, x_n]$ and a field L containing $\mathbb{F}$, by $V(I)(L)$ we mean the set of points $a := (a_1, \cdots, a_n)$ such that a is a zero of all polynomials belonging to I and all the $a_i \in L$. We let $V(I)$ denote the *affine variety* defined by I over $\mathbb{F}$. This is a geometric object with a natural structure of a topological space, where the closed subsets are $V(J)$ for ideals $J \subseteq \mathbb{F}[x_1, \ldots, x_n]$ containing I. This is called the *Zariski topology*. The algebraic set $V(I)(L)$ consists of the L-points of the affine variety $V(I)$.

$$V(I)(L) := \{a = (a_1, \ldots, a_n) \in L^n \mid \forall f \in I,\ f(a) = 0\ \}.$$

On the other hand, given a subset S of $\overline{\mathbb{F}}^n$, let us define $I(S)$ to be the set of polynomials $f \in \overline{\mathbb{F}}[x_1, \ldots, x_n]$ such that $f(s) = 0$, $\forall s \in S$. It is easy to see that $I(S)$ is an ideal of

$\overline{\mathbb{F}}[x_1, \ldots, x_n]$. Let us define:

$$\sqrt{I} := \{f \in \overline{\mathbb{F}}[x_1, \cdots, x_n] \mid \exists\, m \in \mathbb{N} \text{ such that } f^m \in I\}.$$

$\sqrt{I}$ is called the radical of the ideal I. We then have the following basic theorems.

Theorem B.1 (Hilbert's Nullstellensatz). *Let I be an ideal of $\overline{\mathbb{F}}[x_1, \ldots, x_n]$, then:*

$$\sqrt{I} = I(V(I)).$$

We will always deal with radical ideals, namely those I which are equal to $\sqrt{I}$. The affine variety $V(I)$ is often interchangeably used with its $\overline{\mathbb{F}}$-valued points $V(I)(\overline{\mathbb{F}})$, which is the algebraic set it defines.

Given a subset S of $\mathbb{F}^n$, the Zariski-closure of S to be denoted by $Z(S)$ or $\overline{S}$ is by definition the smallest *algebraic* subset of $\mathbb{F}^n$ containing S that is defined by a set of polynomials with coefficients in $\mathbb{F}$.

We call an algebraic subset S *irreducible* if it cannot be written as a union of two algebraic sets S_1 and S_2 properly contained in S. Note that X is irreducible if and only if $I(X)$ is a prime ideal.

A morphism $\phi : X \subseteq \mathbb{A}^n \to \mathbb{A}^1$ from an affine closed subvariety of affine n-space to the affine line is a polynomial map $(x_1, \ldots x_n) \mapsto p(x_1, \ldots, x_n)$, where p is a polynomial. We naturally extend this to a morphism between affine varieties.

Definition B.2 (morphism). *Let $X \subseteq \mathbb{A}^n$ and $Y \subseteq \mathbb{A}^m$ be two closed affine varieties. A morphism $\phi : X \to Y$ is defined to be a map ϕ whose components are polynomials. In other words, ϕ has the form:*

$$\phi(x_1, \ldots, x_n) = (f_1(x_1, \ldots, x_n), \ldots, f_m(x_1, \ldots, x_m)),$$

where $f_1, \ldots f_m$ are polynomials, and with the property that ϕ maps the subset X to Y.

The morphism ϕ is called dominant *if $\phi(X)$ is dense in Y.*

Dimension of a Variety

Let X be a closed affine variety of $\mathbb{A}^n$ over the field $\mathbb{F}$ associated with the ideal $I(X)$. Let $\mathbb{F}(X)$ denote the ring of fractions of the coordinate ring $R = \mathbb{F}[x_1, \ldots, x_n]/I$. If $I(X)$ is a prime ideal, $\mathbb{F}(X)$ is a field and is called the *function field* of X. Elements of the function field $F(X)$ are called *rational functions* on the variety X. In informal terms, the dimension of X is the number of independent rational functions on X.

Definition B.3. *Let $\mathbb{K}$ be a finitely generated extension field over a base field $\mathbb{F}$. Let S be a maximal set of algebraically independent elements of $\mathbb{K}$ over $\mathbb{F}$. Such an S is called a* transcendence basis *of $\mathbb{K}$ over $\mathbb{F}$. It can be proved that $|S|$ is independent of S, and is called the* transcendence degree *of $\mathbb{K}$ over F and will be denoted by $tr.deg(K/F)$.*

Definition B.4. *The dimension of an affine variety $X \subseteq F^n$ denoted by $\dim(X)$ is the transcendence degree of the function field $\mathbb{F}(X)$ of the variety X over the base field $\mathbb{F}$. Thus, $\dim(X) := tr.deg(F(X)/F)$.*

For easy reference we state a lemma below that we need.

Lemma B.5 ([Sha94a])**.** *Let $\phi : X \to Y$ be a dominant morphism. Then ϕ^* induces a natural isomorphic inclusion of $\mathbb{F}(Y) \hookrightarrow \mathbb{F}(X)$. In particular, $\dim(Y) = tr.deg(F(Y)) \leq tr.deg(F(X) = dim(X)$.*

To give a more intuitive understanding of the notion of dimension, we describe a simple example. If V is a linear subspace of $\mathbb{F}^n$ (or a translate of such a subspace) of dimension d (in the linear algebraic sense), then it is an easy to show that the dimension of V as an affine variety is also d. Indeed, one can (upto isomorphism) choose a set of d coordinates $x_{i_1}, \ldots, x_{i_d}$ and variables corresponding to the remaining coordinates (as polynomials of degree 1) form the ideal defining the variety. In other words, the coordinate ring $\mathbb{F}[V] = \mathbb{F}[x_1, \ldots, x_n]/I(V)$ is canonically isomorphic to $\mathbb{F}[x_{i_1}, \ldots, x_{i_d}]$ (where the x_{i_j} are the *free* variables in the system of linear equations defining V). Thus, the dimension of $\mathbb{F}(V)$ over $\mathbb{F}$ is exactly d.

In more intuitive terms, in the case of linear algebra, we say V has dimension n by pointing out that its elements are parametrised by n-tuples. However, in the case of algebraic sets, it is not true, in general, that the points of an algebraic set of dimension n are parametrised by n-tuples; the most one can say is that *for any irreducible algebraic variety of dimension d, there is a finite surjective map $\phi : V \to \mathbb{A}^d$.* This is a re-statement of the *Noether Normalization Theorem*.

A related question to ask is how many polynomials (equations) are needed to define an algebraic set V. If V is a linear subspace of $\mathbb{F}^n$(or translate of such a subspace), then linear algebra shows that it is the zero set of $n - \dim(V)$ polynomials. But in general, for an algebraic set, all one can say is that at least $n - \dim(V)$ polynomials are needed to define

V. However, in most cases, many more are required. Given a variety, determining exactly how many is an area of active research.

Gröbner basis

Let the set $\mathbb{T}$ of products of the variables be defined as follows

$$\mathbb{T}^n = \left\{ x_1^{\beta_1} \cdots x_n^{\beta_n} \mid \beta_i \in \mathbb{N}, i = 1, \dots n \right\}$$

Sometimes we will denote $x_1^{\beta_1} \cdots x_n^{\beta_n}$ by x^β, where $\beta = (\beta_1, \dots, \beta_n) \in \mathbb{N}^n$. A term order is a total order on the set $\mathbb{T}^n$ that is a well ordering. A simple example is the lexicographical ordering. Fix some term order, on $\mathbb{K}[x_1, \dots, x_n]$. Then for all $f \in \mathbb{K}[x_1, \dots x_n]$, with $f \neq 0$, we may write it as $f = a_1 x^{\alpha_1} + a_2 x^{\alpha_2} + \cdots + a_r x^{\alpha_r}$, where for all i, a_i's are non-zero and $x^{\alpha_i} > x^{\alpha_{i+1}}$. Now $lp(f) = x^{\alpha_i}$.

Definition B.6 (Gröbner basis)**.** *The Gröbner basis of an ideal I is the set of polynomials $G = \{g_1, \dots, g_k\}$ contained in I such that for all $f \in I$ such that $f \neq 0$, there exists $i \in \{1, \dots, t\}$ for which the leading power term (w.r.t. the chosen term order) of g_i divides that of f; or equivalently, $Lt(I) = Lt(G)$. In addition, if for all i no non-zero term in g_i is divisible by any $lp(g_j)$ for $i \neq j$, then we say that the basis is a* reduced *Gröbner basis.*

Gröbner basis need not be unique. Moreover, it also depends on the choice of the term order. Buchberger [Buc83] proved that for fixed a term order, every non-zero ideal has a unique reduced Gröbner basis with respect to this term order. He also suggested an algorithm to compute the reduced Gröbner basis.

Transversality and Tangent Spaces

A (topological) manifold is an abstract mathematical space in which every point has a neighborhood which resembles Euclidean space, $\mathbb{R}^n$. Formally, it is a topological space locally homeomorphic to a Euclidean space. This means that every point has a neighbourhood for which there exists a homeomorphism (a bijective continuous function whose inverse is also continuous) mapping that neighbourhood to an open subset of $\mathbb{R}^n$.

In a one-dimensional manifold (or one-manifold), every point has a neighborhood that looks like a segment of a line. Examples of one-manifolds are a line, a circle, two separate circles etc. In a two-manifold, every point has a neighborhood that looks like a disk.

Examples include a plane, the surface of a sphere, and the surface of a torus etc. Algebraic geometers view an affine variety as a topological object, and study topological notion of tangent spaces.

Now, intuitively, a subspace is a tangent to a curve at a point if it is closest approximation to the neighbourhood at that point. We can make this a little more formal.

Let $p \in \mathbb{V} \subset \mathbb{A}^n$ be a point on an affine variety defined by $f_1 = f_2 = \dots f_\ell = 0$. The tangent space at $p = (p_1, \dots p_n)$, denoted by T_pV, is the subspace of the vector space with origin a cut out by the set of ℓ linear equations given by: $1 \leq j \leq \ell$,

$$\sum_{i=1}^{n} \frac{\partial f_j}{\partial x_i}(p)(x_i - p_i) = 0$$

Thus, $\dim T_pV = n - rank(J)$, where J is the *Jacobian matrix* defined by,

$$\begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_\ell}{\partial x_1} \\ \vdots & \cdots & \vdots \\ \frac{\partial f_1}{\partial x_n} & \cdots & \frac{\partial f_\ell}{\partial x_n} \end{pmatrix}$$

Points on the variety where $\dim T_pV = \dim V$ are called *non-singular or smooth points*. Points where $\dim T_pV \neq \dim V$ are called *singular points*. If V us reducible a point p is non-singular if $\dim T_pV$ is equal to the maximum dimension of an irreducible component passing through p. In this case it turns out that the singular points are exactly those points which either (1) lie in more than one component (2) is a singular point of the unique irreducible variety in which it lies.

Two subvarieties of a given variety are said to intersect *transversally* if at every point of intersection, their separate tangent spaces at that point together generate the tangent space of the variety at that point.

For sets $A, B, C \subseteq \mathbb{A}^k$. If we have an additive map $A \times B$ onto C as a well defined affine map $\phi : \mathbb{A}^k \times \mathbb{A}^k \rightarrow \mathbb{A}^k$; to prove that $\dim C = \dim A + \dim B$, it is sufficient to find a smooth point of $A \times B$ for which the map ϕ acts injectively on the corresponding tangent spaces. If we verify that the two tangent spaces intersect transversally, it proves that dimension of the set C is at least the sum of that of A and B. This abstract idea is used in proof of Theorem 7.8.

Appendix C

Algebraic Number Theory Preliminaries

Although the following can be stated in general for any field, the fields that we consider will be $\mathbb{Q}$ and extensions of $\mathbb{Q}$ since we require only that in the proof of Theorem 7.10. We refer the reader to [AW04] for a more general treatment.

Suppose $\mathbb{K}/\mathbb{Q}$ is a field extension (which means $\mathbb{K}$ is a field and $\mathbb{Q}$ is a subfield of $\mathbb{K}$). We call $\mathbb{K}$ is finite extension of $\mathbb{Q}$, if $\mathbb{K}$ is a finite dimensional vector space over $\mathbb{Q}$. The dimension of this vector space is the degree of the extension. For our case all extensions will be finite. Finite extensions of $\mathbb{Q}$ are called *algebraic number fields*.

For $\alpha_1, \ldots, \alpha_n \in \mathbb{K}$ we can think of the smallest subfield of $\mathbb{K}$ containing $\mathbb{Q}$ and these elements, denoted by $\mathbb{Q}(\alpha_1. \ldots, \alpha_n)$. When there exists αs such that $\mathsf{L} = \mathbb{Q}(\alpha_1. \ldots, \alpha_n)$, the the extension is *finitely generated*.

An element $\alpha \in \mathbb{K}$ is *algebraic* over $\mathbb{Q}$ if there is a non-zero polynomial with coefficients in $\mathbb{Q}$ which it satisfies. If an element α is algebraic, the non-zero monic polynomial of least degree (which by definition is irreducible), p_α, which it satisfies, is called the *minimal polynomial* of α over $\mathbb{Q}$. The extension is algebraic if every element of L is algebraic over $\mathbb{Q}$.

Now we talk about algebraic extensions. The extension is *normal* if and only if for every $\alpha \in \mathsf{L}$, all the roots of p_α are in L. In other words, L contains the splitting field of every $\alpha \in L$. If an extension is finite and normal, then it is simple. That is, there exists a single element $\alpha \in \mathsf{L}$ such that $\mathbb{K} = \mathbb{Q}(\alpha)$

The extension is *separable* if for every $\alpha \in \mathsf{L}$, all the roots of p_α are distinct. An extension is *Galois* if it is normal and separable. An element $\alpha \in \mathbb{K}$ is an algebraic integer if its minimal polynomial p_α is monic with integral coefficients. The set of algebraic integers form a ring, called *ring of integers*.

Having defined algebraic numbers it is a natural question to ask if the unique factorisation property enjoyed by the integers also holds in the ring of integers in an algebraic number field. In 1844, E. Kummer showed that this does not hold, in general. About three years later, he showed that the unique factorisation in such rings is possible if numbers are replaced by the notion of *ideal numbers*. In effect, Dedekind showed that the ring of integers of an algebraic number field has the following property: *every nonzero ideal in this ring factors uniquely as a product of prime ideals*.

Now in the ring of integers of a number field, a prime $p \in \mathbb{Z}$ may not remain a prime. For instance, in $\mathbb{Q}(\sqrt{-1})$, 2 and 5 are no longer prime numbers but 3 is. This can be seen as But the ideal generated by p, can be uniquely factored into prime ideals. Roughly speaking, the phenomenon of a prime splitting into several primes in an extension, is known as ramification. A prime number $p \in \mathbb{Z}$ (also called the *rational prime*) is said to be totally ramified in an algebraic number field of degree n, if the ideal generated by p, denoted by $\langle p \rangle$ is the power of some prime ideal in the ring of integers $\mathcal{O}_{\mathbb{K}}$. The prime is said to be unramified if in the prime decomposition of the corresponding ideal, every prime ideal occurs at most once.

Now, by choice of $\zeta_\alpha = e^{2\pi \mathrm{i}/p_\alpha}$, we know that p_α totally ramifies in $\mathbb{Q}(\zeta_\alpha)$. It is also true that it is unramified in $\mathbb{Q}(\zeta_1, \ldots, \zeta_{\alpha-1}, \zeta_{\alpha+1}, \ldots, \zeta_{n^2})$. Thus the intersection of these two extensions cannot contain anything more than the base field $\mathbb{Q}$. Thus,

$$\mathbb{Q}(\zeta_\alpha) \cap \mathbb{Q}(\zeta_1, \ldots, \zeta_{\alpha-1}, \zeta_{\alpha+1}, \ldots, \zeta_{n^2}) = \mathbb{Q}.$$

Bibliography

[AAB$^+$99] E. Allender, A. Ambainis, D. A. Mix Barrington, S. Datta, and H. LeThanh. Bounded-depth Arithmetic Circuits: Counting and Closure. In *Proceedings of 26th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 1644 of *Lecture Notes in Computer Science*, pages 149–158. Springer-Verlag, 1999.

[AAM03] E. Allender, V. Arvind, and M. Mahajan. Arithmetic Complexity, Kleene Closure, and Formal Power Series. *Theory of Computing Systems*, 36(4):303–328, 2003.

[ABC$^+$06] E. Allender, D. A. Mix Barrington, T. Chakraborty, S. Datta, and S. Roy. Grid Graph Reachability Problems. In *Proceedings of 21st IEEE conference on computational complexity (CCC 2006)*, pages 299–313, 2006.

[ABO96] Eric Allender, Robert Beals, and Mitsunori Ogihara. The Complexity of Matrix Rank and Feasible Systems of Linear Equations. In *Proc. 28th ACM Symposium on Theory of Computing (STOC 96)*, pages 161–167, 1996. Journal version : Computational Complexity 8(2), 99–126, 1999.

[ABSS93] Sanjeev Arora, L Babai, Jacques Stern, and Z. Sweedyk. The Hardness of Approximate Optimia in Lattices, Codes, and Systems of Linear Equations. In *IEEE Symposium on Foundations of Computer Science,* pages 724–733, 1993.

[ADR05a] E. Allender, S. Datta, and S. Roy. The Directed Planar Reachability Problem. In *Proceedings of 25th International Conference Foundations of Software Technology and Theoretical Computer Science(FSTTCS)*, volume 3821 of *Lecture Notes in Computer Science,* pages 238–249, 2005.

[ADR05b] Eric Allender, Samir Datta, and Sambuddha Roy. Topology inside NC^1. In *Proceedings of IEEE Conference on Computational Complexity (CCC 2005)*, 2005. Electronic Colloquium on Computational Complexity (ECCC), TR04-108.

[AG00] C Àlvarez and R Greenlaw. A compendium of problems complete for symmetric logarithmic space. *Computational Complexity*, 9:73–95, 2000.

[All04] E. Allender. Arithmetic Circuits and Counting Complexity Classes. In Jan Krajicek, editor, *Complexity of Computations and Proofs*, Quaderni di Matematica Vol. 13, pages 33–72. Seconda Universita di Napoli, 2004. An earlier version appeared in the Complexity Theory Column, SIGACT News 28, 4 (Dec. 1997) pp. 2-15.

[Alo94] N. Alon. On the Rigidity of a Hadamard matrix. Manuscript, 1994.

[AM04] E. Allender and M. Mahajan. The Complexity of Planarity Testing. *Information and Computation*, 189(1):117–134, 2004.

[AO96] E. Allender and M. Ogihara. Relationships among PL, #L, and the Determinant. *RAIRO Theoretical Information and Applications*, 30:1–21, 1996. Conference version in *Proc. 9th IEEE Structure in Complexity Theory Conference* (1994), 267–278.

[AW04] Saban Alaca and Kenneth S. Williams. *Introductory Algebraic Number Theory*. Cambridge University Press., 2004.

[AW08] Scott Aaronson and Avi Wigderson. Algebrization: A New Barrier in Complexity Theory. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, 2008.

[Bar75] Stephen Barnett. *Introduction to Mathematical Control Theory*. Oxford Applied Mathematics and Computing Science. Oxford University Press, 1975.

[Bar89] David A. Mix Barrington. Bounded-width Polynomial-size Branching Programs Recognize Exactly Those Languages in NC^1. *Journal of the Computer and System Sciences*, 38:150–164, 1989. Preliminary version appeared in Proceedings of the 18th Annual ACM Symposium on Theory of Computing (1986).

[BBF$^+$74] J.R. Barclay, J.D. Bransford, J.J. Franks, N.S. MacCarell, and K. Nitsch. Comprehension and Semantic Flexibility. *Journal of Verbal Learning and Verbal Behaviour*, 13:471–481, 1974.

[BBF03] C. Bachmaier, F.-J. Brandenburg, and M. Forster. Radial Level Planarity Testing and Embedding in Linear Time. In *Proceedings of 11th International Symposium on Graph Drawing (GD 03), Perugia, Italy*, volume 2912 of *Lecture Notes in Computer Science*, pages 393–405, 2003.

[BBMT98] P. Bertolazzi, G. Di Battista, C. Manning, and R. Tamassia. Optimal Upward Planarity Testing of Single-source Digraphs. *SIAM Journal on Computing*, 27:132–169, 1998.

[BCS97] Peter Bürgisser, Michael Clausen, and Mohammad A. Shokrollahi. *Algebraic Complexity Theory*, volume 315 of *Grundlehren der mathematischen Wissenschaften*. Springer Verlag, 1997.

[BDHM92] G. Buntrock, C. Damm, U. Hertrampf, and C. Meinel. Structure and Importance of Logspace MOD-classes. *Math. Systems Theory*, 25:223–237, 1992.

[BDO95] M. Berry, S. Dumais, and G. O'Berie. Using Linear Algebra for Intelligent Information Retrieval. *SIAM Review*, 37:573–595, 1995.

[BFS86] László Babai, Peter Frankl, and Janos Simon. Complexity Classes in Communication Complexity Theory (preliminary version). In *Proceedings of 27th Annual Symposium on Foundations of Computer Science, 27-29 October 1986, Toronto, Ontario, Canada*, pages 337–347. IEEE, 1986.

[BFS97] Jonathan F. Buss, Gudmund Skovbjerg Frandsen, and Jeffrey Shallit. The Computational Complexity of Some Problems of Linear Algebra (Extended Abstract). In *Proceedings of 16th Annual Symposium on Theoretical Aspects of Computer Science(STACS)*, volume 1200 of *Lecture Notes in Computer Science*, pages 451–462, 1997. JCSS 1999, 58 pp 572-596.

[BGS75] Theodore P. Baker, John Gill, and Robert Solovay. Relativizatons of the P vs. NP Question. *SIAM Journal of Computing*, 4(4):431–442, 1975.

[BI97] David Mix Barrington and Neil Immerman. Time, Hardware, and Uniformity. In *Complexity theory retrospective II*, chapter 1, pages 1–22. Springer-Verlag New York, Inc., 1997.

[Bie90] Daniel Bienstock. Some provably hard crossing number problems. In *SCG '90: Proceedings of the sixth annual symposium on Computational geometry*, pages 253–260, 1990.

[BK79] F. Bernhart and P. C. Kainen. The Book Thickness of a Graph. *Journal of Combinatorial Theory, Ser. B*, 27(3):320–331, 1979.

[BKR07] Mark Braverman, Raghav Kulkarni, and Sambuddha Roy. Parity Problems in Planar Graphs. In *IEEE Conference on Computational Complexity*, pages 222–235, 2007.

[BLMS97] David A. Mix Barrington, Chi-Jen Lu, Peter Bro Miltersen, and Sven Skyum. Searching Constant Width Mazes Captures the AC^0 Hierarchy. In *Proceedings of Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 73–83, 1997. ECCC Technical report TR97-044.

[BLMS99] David A. Mix Barrington, Chi-Jen Lu, Peter Bro Miltersen, and Sven Skyum. On Monotone Planar Circuits. In *Proceedings of the 14th Annual IEEE Conference on Computational Complexity (CCC)*, pages 24–31, 1999.

[BMMR02] Anna Bernasconi, Earnst W. Mayr, Michal Mnuk, and Martin Raab. Computing the Dimension of a Polynomial Ideal. http://www14.informatik.tu-muenchen.de/personen/raab/, 2002.

[Bol70] V.G. Boltyanskii. *Mathematical Methods of Optimal Control*. Holt, Rinehard and Wineston Inc., 1970. Tranlated from Russian by K.N. Tririgoff.

[Bol84] B. Bollobas. *Modern Graph Theory*, volume 184 of *Graduate texts in mathematics*. Springer, 1984.

[Bro87] W. D. Brownawell. Bounds on the degrees of Nullstellensatz. *Annals of Mathematics*, 126:577–592, 1987.

[BT88] G. Di Battista and R. Tamassia. Algorithms for plane representations of acyclic digraphs. *Theoretical Computer Science*, 61:175–198, 1988.

[BT00] Vincent D. Blondel and John N. Tsitsiklis. A Survey of Computational Complexity Results in Systems and Control. *Automatica*, 36(9):1249–1274, 2000.

[Buc83] B. Buchberger. A Note on the Complexity of Constructing Grobner Bases. In *EUROCAL '83, European Computer Algebra Conference*, pages 137–145, 1983.

[Bus87] S. Buss. The Boolean Formula Value Problem Is in ALOGTIME. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 123–131, 1987.

[BV80] Winfried Bruns and Udo Vetter. *Determinantal Rings*, volume 1327 of *Lecture Notes in Mathematics*. Springer-Verlag, 1980.

[CD06] Tanmoy Chakraborty and Samir Datta. One-input-face MPCVP is hard for L, but in LogDCFL. In S. Arun Kumar and Naveen Garg, editors, *Proc. of 26th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 06)*, volume 4337 of *Lecture Notes in Computer Science*, pages 57–68, 2006.

[Che05] Mahdi Cheraghchi. On Matrix Rigidity and the Complexity of Linear Forms. *Electronic Colloquium on Computational Complexity (ECCC)*, (070), 2005.

[Chi85] Alexander L. Chistov. Fast Parallel Computation of the Rank of Matrices over a Field of Arbitrary Characteristic. In *Proceedings of 10th International Symposium on Fundamentals of Computation Theory (FCT'95)*, volume 199 of *Lecture Notes in Computer Science*, pages 63–69, 1985.

[CLO07] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties,and Algorithms (An Introduction to Computational Algebraic Geometry and Commutative Algebra)*. Under Graduate Textbooks in Mathematics. 3rd edition, 2007.

[CM87] Stephen Cook and Pierre McKenzy. Problems Complete for Deterministic Logarithmic Space. *Journal of Algorithms*, 8:385–394, 1987.

[CMTV98] Herve Caussinus, Pierre McKenzie, Denis Therien, and Heribert Vollmer. Nondeterministic NC^1 Computation. *Journal of Computer and System Sciences*, 57(2):200–212, 1998.

[Cod00] Bruno Codenotti. Matrix Rigidity. *Linear Algebra and its Applications*, 304(1–3):181–192, 2000.

[Coo71] S. Cook. Characterizations of pushdown machines in terms of time-bounded computers. *Journal of Association of Computing Machinery*, 18:4–18, 1971.

[Coo03] Stephen Cook. The Importance of the P versus NP question. *Journal of the ACM*, 50(1):27–29, 2003.

[Csa76] L. Csanky. Fast parallel matrix inversion algorithms. *SIAM Journal on Computing*, 5(4):618–623, 1976.

[Dah99] G. Dahl. A note on nonnegative diagonally dominant matrices. *Linear Algebra and Applications*, 317:217–224, 1999.

[Dam91] C. Damm. $det = l(\#l)$. Technical Report Informatik-Preprint 8, Fachbereich Informatik der Humboldt–Universität zu Berlin, 1991.

[DC89] P. W. Dymond and S. A. Cook. Complexity Theory of Parallel Time and Hardware. *Information and Computation*, 80(3):205–226, 1989.

[DEH00] M. B. Dillencourt, D. Eppstein, and D.S. Hirschberg. Geometric thickness of complete graphs. *Journal of Graph Algorithms and Applications*, 4(3):5–17, 2000.

[Dem92] James Demmel. The componentwise distance to the nearest singular matrix. *SIAM Journal on Matrix Analysis and Applications*, 13(1):10–19, 1992.

[Des07] Amit Deshpande. *Sampling-based dimension reduction algorithms*. PhD thesis, MIT, May 2007.

[DK95] A. L. Delcher and S. R. Kosaraju. An NC algorithm for Evaluating Monotone Planar Circuits. *SIAM Journal of Computing*, 24(2):369–375, 1995.

[DK00] Ding-Zhu Du and Ker-I Ko. *Theory of Computational Complexity*. Wiley-Interscience series in Discrete Mathematics and Optimisation. Wiley Interscience Publication, 2000.

[DM01] I. Dhillon and D. Modha. Concept Decompositions for Large Sparse Text Data Using Clustering. *Machine Learning*, 42:143–175, 2001.

[DV07] Amit Deshpande and Kasturi R. Varadarajan. Sampling-based dimension reduction for subspace approximation. In *STOC*, pages 641–650, 2007.

[Edm65] Jack Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965.

[Epp01] David Eppstein. Separating geometric thickness from book thickness. arXiv.org:math/0109195, 2001.

[EY36] Carl Eckart and Gale Young. The Approximation of One Matrix by Another of Lower Rank. *Psychometrika*, 1(3):211–218, September 1936.

[FH03] Lance Fortnow and Steven Homer. A short history of computational complexity. *Bulletin of the EATCS*, 80:95–133, 2003.

[Fis74] Michael J. Fischer. The Complexity of Negation-limited Networks (a brief survey). *Lecture Notes in Computer Science*, 33:71–82, 1974.

[FKL$^+$01] Jürgen Forster, Matthias Krause, Satyanarayana V. Lokam, Rustam Mubarakzjanov, Niels Schmitt, and Hans Ulrich Simon. Relations between communication complexity, linear arrangements, and computational complexity. In *Fst tcs 2001: foundations of software technology and theoretical computer science (bangalore)*, volume 2245 of *Lecture Notes in Comput. Sci.*, pages 171–182. Springer, Berlin, 2001.

[FKV04] Alan M. Frieze, Ravi Kannan, and Santosh Vempala. Fast monte-carlo algorithms for finding low-rank approximations. *Journal of ACM*, 51(6):1025–1041, 2004.

[For02] Jürgen Forster. A linear lower bound on the unbounded error probabilistic communication complexity. *J. Comput. System Sci.*, 65(4):612–625, 2002. Special issue on complexity, 2001 (Chicago, IL).

[Fri93] J. Friedman. A Note on matrix rigidity. *Combinatorica*, 13(2):235–239, 1993.

[Gee99] James F. Geelen. Maximum rank matrix completion. *Linear Algebra and its Applications*, 288(1–3):211–217, 1999.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H.Freeman and Co., 1979.

[GJ92] M.R. Garey and D.S. Johnson. The np-completeness column: An ongoing guide. *Journal of Algorithms*, 3:89–99, 1992.

[Gol77] L. M. Goldschlager. The monotone and planar circuit value problems are logspace complete for P. *SIGACT News*, 9(2):25–29, 1977.

[Gol80] Leslie M. Goldschlager. A Space Efficient Algorithm for the Monotone Planar Circuit Value Problem. *Information Processing Letters*, 10(1):25–27, 1980.

[Gri76] Dima Grigoriev. Using the Notions of Seperability and Independence for Proving the Lower Bounds on the Circuit Complexity (In Russian). Notes of the Leningrad branch of the Steklov Mathematical Institute, Nauka, 1976. English translation in Journal of Soviet Math., 14(5), pp. 1450-1456, 1980.

[GT01] A. Garg and R. Tamassia. On the computational complexity of upward and rectilinear planarity testing. *SIAM Journal on Computing*, 31(22):601–625, 2001.

[Han04] Kristoffer Arnsfelt Hansen. Constant Width Planar Computation Characterizes ACC^0. In *21st Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 2996 of *Lecture Notes in Computer Science*, pages 44–55, 2004. Journal Version : Theory of Computing Systems, 39(1), 79-92, 2006.

[Han08] Kristoffer Arnsfelt Hansen. Constant Width Planar Branching Programs Characterize ACC^0 in Quasipolynomial Size. In *IEEE Conference on Computational Complexity (CCC 2008)*, 2008. To appear.

[HE71] M. Hochster and J.A. Eagon. Cohen-Macaulay Rings, Invariant Theory, and the generic perfection of determinantal loci. *American Journal of Mathematics*, 93:1020–1058, 1971.

[HMV06] Kristoffer Arnsfelt Hansen, Peter Bro Miltersen, and V Vinay. Circuits on cylinders. *Computational Complexity*, 15(1):62–81, May 2006.

[Hås86] Johan Håstad. Almost Optimal Lower Bounds for Small Depth Circuits. In *Proceedings of the 18th Annual ACM Symposium on Theory of computing : STOC '86*, pages 6–20, 1986.

[IL89] Neil Immerman and Susan Landau. The complexity of iterated multiplication. In *Structure in Complexity Theory Conference*, pages 104–111, 1989.

[IMR80] Oscar H. Ibarra, Shlomo Moran, and Louis E. Rosier. A note on the parallel complexity of computing the rank of order n matrices. *Inf. Process. Lett.*, 11(4/5):162, 1980.

[JLL76] Neil D. Jones, Y.E. Lien, and William T. Laaser. New problems complete for non-deterministic logspace. *Mathematical Systems Theory*, 10:1–17, 1976.

[Kai73] P. C. Kainen. Thickness and coarseness of graphs. *Abh. Math. Sem. Univ. Hamburg*, 39:88–95, 1973.

[Kel87] D. Kelly. Fundamentals of planar ordered sets. *Discrete Mathematics*, 63(2,3):197–216, 1987.

[Kol88] Janos Kollar. Sharp Effective Nullstellensatz. *Journal of American Mathematical Society*, 1(4):963–975, 1988.

[Kos90] S. Rao Kosaraju. On the Parallel Evaluation of Classes of Circuits. In *Proc. 10th FSTTCS Conference, LNCS vol. 472*, pages 232–237, 1990.

[KR97] Boris S. Kashin and Alexander A. Razborov. Improved lower bounds on the rigidity of hadamard matrices (in Russian). *Matematicheskie Zametki*, 63(4):535–540, 1997. English translation available at authors webpage.

[Kul07] Raghav Kulkarni. Personal communication, January 2007.

[Lan04] Serge Lang. *Algebra*. Springer-Verlag, revised third edition, 2004.

[Lau01] M. Laurent. Matrix completion problems. In C.A. Floudas and P.M. Pardalos, editors, *The Encyclopedia of Optimization*, volume 3, pages 221–229. Kluwer, 2001.

[Lew06] Andrew C. Lewis. Semicontinuity of Rank and Nullity and Some Consequences, January 2006. Manuscript.

[Li92] L. Li. Formal power series: An algebraic approach to the GapP and #P functions. In *Proc. 7th Structure in Complexity Theory Conference*, pages 144–154, 1992.

[LMN08] Nutan Limaye, Meena Mahajan, and Prajakta Nimbhorkar. Longest Paths in Planar DAGs in Unambiguous Logspace. http://arxiv.org/abs/0802.1699, February 2008.

[Lok95] Satyanarayana V. Lokam. Spectral methods for matrix rigidity with applications to size-depth tradeoffs and communication complexity. In *Proc. 36th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 6 – 15, 1995. Journal of Computer and System Sciences, 63(3):449-473, 2001.

[Lok00] Satyanarayana V. Lokam. On the Rigidity of Vandermonde matrices. *Theoret. Comput. Sci.*, 237(1-2):477–483, 2000. Presented at the DIMACS-DIMATIA workshop on Arithmetic Circuits and Algebraic Methods, June , 1999.

[Lok06] Satyanarayana V. Lokam. Quadratic Lowerbounds on Matrix Rigidity. In *Proceedings of International Conference on Theory and Applications of Models of Computation (TAMC 2006)*, volume 3959 of *Lecture Notes in Computer Science*, 2006.

[Lov75] Lászlo Lovász. On the ratio of optimal integral and fractional covers. *Discrete Mathematics*, 13(4):383–390, 1975.

[LS06] Nathan Linial and Adi Shreibman. Learning Complexity vs. Communication Complexity. Weblink : http://www.cs.huji.ac.il/ nati/PAPERS/lcc.pdf, 2006.

[LTV03] J. M. Landsberg, J. Taylor, and Nisheeth K. Vishnoi. The Geometry of Matrix Rigidity. Technical Report GIT-CC-03-54, Georgia Institute of Technology, http://smartech.gatech.edu/handle/1853/6514, 2003.

[Lyp58] O.B. Lypanov. On the Synthesis of Contact Networks. *Dokl. Akad. Nauk. SSSR*, 119:23–26, 1958.

[Mar58] A. A. Markov. On the Inversion Complexity of a System of Functions. *The Journal of the Association of Computing Machinery (JACM)*, 5(4):331–334, 1958.

[MM64] Marvin Marcus and Henryk Minc. *A Survey of Matrix Theorey and Matrix inequalities*, volume 14 of *The Prindle, Weber and Schmidt Complementary Series in Mathematics*. Allyn and Bacon, Boston, 1964.

[MOS98] P. Mutzel, T. Odenthal, and M. Scharbrodt. The thickness of graphs: a survey. *Graphs Combinatorics*, 14(1):59–73, 1998.

[MRK88] G.L. Miller, V. Ramachandran, and E. Kaltofen. Efficient parallel evaluation of straight-line code and arithmetic circuits. *SIAM Journal of Computing*, 17:687–695, 1988.

[MRV99] P. McKenzie, K. Reinhardt, and V. Vinay. Circuits and context-free languages. In *Proceedings of 5th Annual International Computing and Combinatorics Conference (COCOON)*, 1999.

[MS07] Ketan D. Mulmuley and Milind Sohoni. Geometric Complexity Theory : An Introduction. Technical Report TR-2007-16, Computer Science Department, University of Chicago, 16 October 2007.

[MT01] B. Mohar and C. Thomassen. *Graphs on Surfaces*. John Hopkins University Press, Maryland, 2001.

[Mul87] Ketan Mulmuley. A Fast Parallel Algorithm to Compute the Rank of a Matrix over an Arbitrary Field. *Combinatorica*, 7:101–104, 1987.

[Mur93] K. Murota. Mixed matrices - Irreducibility and decomposition. In R. A. Brualdi, S. Friedland, and V. Klee, editors, *Combinatorial and Graph-Theoretic Problems in Linear Algebra*, volume 50 of *The IMA Volumes in Mathematics and Its Applications*, pages 39 – 71. Springer, 1993.

[MV97] M. Mahajan and V Vinay. Determinant: combinatorics, algorithms, complexity. *Chicago Journal of Theoretical Computer Science*, 1997:5, dec 1997.

[Nar04] Wladyslow Narkiewicz. *Elementary and Analytic Theory of Algebraic Numbers*, volume XI of *Springer Monographs in Mathematics*. Springer, 2004.

[NTS95] N. Nisan and A. Ta-Shma. Symmetric Logspace is closed under complement. *Chicago Journal of Theoretical Computer Science*, 1995.

[Pip79] Nicholas Pippenger. On Simultaneous Resource Bounds. In *20th IEEE Foundations of Computer Science (FOCS 1979)*, pages 307–311, 1979.

[PP04] Ramamohan Paturi and Pavel Pudlák. Circuit Lower Bounds and Linear Codes. In E. A. Hirsch, editor, *Notes of Mathematical Seminars of St.Petersburg Department of Steklov Institute of Mathematics*, volume 316 of *Teoria slozhnosti vychislenij IX*, pages 188–204, 2004. Technical Report appeared in ECCC : TR04-04.

[PR93] S. Poljak and J. Rohn. Checking robust nonsingularity is NP-hard. *Math. Control Signals Systems*, 6:1–9, 1993.

[Pud94] Pavel Pudlak. Communication in bounded depth circuits. *Combinatorica*, 14(2):203–216, 1994.

[PV91] P. Pudlak and Z. Vavrin. Computation of rigidity of order n^2/r for one simple matrix. *Comment. Nath. Univ. Carolinae.*, 32(2):213–218, 1991.

[Raz87] Alexander A. Razborov. Lower bounds on the size of Bounded Depth Networks over a Complete Basis with Logical Addition. *Mathematicheskie Zametski*, 41(4):598–607, 1987. English Translation in Mathematical Notes of the Academy of Sciences in USSR 41(4), 333-338.

[Raz88] Alexander A. Razborov. Bounded-depth formulae over the basis {AND, XOR} and some combinatorial problems (Russian). In *Problems of Cybernetics. Complexity Theory and Applied Mathematical Logic*, pages 149–166. 1988.

[Raz89] Alexander A. Razborov. On rigid matrices. manuscript in russian, 1989.

[Rei05] O. Reingold. Undirected st-conenctivity in logspace. In *Proc. 37th Annual ACM Symposium on Theory of Computing STOC*, pages 376–385, 2005.

[Roh89] Jiri Rohn. Systems of Linear Interval Equations. *Linear Algebra and Its Applications,* 126:39–78, 1989.

[Roh94] J. Rohn. Checking positive definiteness or stability of symmetric interval matrices is NP-hard. *Commentationes Mathematicae Universitatis Carolinae*, 35:795–797, 1994.

[Roh96] Jiri Rohn. *Checking Properties of Interval Matrices*. Kluwer, 1996. 36 p.

[Roy06] Sambuddha Roy. *Complexity Theoretic Aspects of Planar Restrictions and Obliviousness*. PhD thesis, Department of Computer Science, Rutgers University, 2006.

[RR94] Alexander A. Razborov and Steven Rudich. Natural proofs. In *ACM Symposium on Theory of Computing (STOC)*, pages 204–213, 1994. Full version appears in Journal of Computer and System Sciences (JCSS) 55(1): 24-35 (1997).

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

[Rum03a] Siegfried M. Rump. Structured Perturbations Part I: Normwise Distances. *SIAM Journal of Matrix Analysis and Applications*, 25(1):1–30, 2003.

[Rum03b] Siegfried M. Rump. Structured Perturbations Part II: Componentwise Distances. *SIAM Journal of Matrix Analysis and Applications*, 25(1):31–56, 2003.

[Ruz80] W.L. Ruzzo. Tree-size bounded alternation. *Journal of Computer and System Sciences*, 21:218–235, 1980.

[RW91] P. Radge and A. Wigderson. Linear-size constant-depth polylog-threshold circuits. *Information Processing Letters*, 39:143–146, 3 1991.

[San07] Rahul Santhanam. Circuit lower bounds for merlin-arthur classes. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC)*, pages 275–283, 2007.

[Sha49] Claude E. Shannon. The Synthesis of Two-terminal Switching Circuits. *Bell Systems Technical Journal*, (28):59–98, 1949.

[Sha94a] Igor R. Shafarevich. *Schemes and Complex Manifolds*, volume 2 of *Basic Algebraic Geometry*. Springer-Verlag New York, Inc., second edition, 1994.

[Sha94b] Igor R. Shafarevich. *Varieities in Projective Space*, volume 1 of *Basic Algebraic Geometry*. Springer Verlag, second edition, 1994.

[SJ03] N. Srebro and T. Jakkola. Weighted Low Rank Approximations. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 720–727, 2003.

[SK92] Amber Settle and Peter Kimmel. Reducing the Rank of Lower Triangular All-Ones Matrices. Technical Report TR-92-21, Department of Computer Science, University of Chicago, 6 November 1992.

[Smo87] Roman Smolensky. Algebraic Methods in the Theory of Lower Bounds for Boolean Circuit Complexity. In *Proceedings of the 19th Annual ACM conference on Theory of Computing (STOC)*, pages 77–82, 1987.

[SS91] Victor Shoup and Roman Smolensky. Lower bounds for polynomial evaluation and interpolation problems. In *IEEE Symposium on Foundations of Computer Science*, pages 378–383, 1991.

[SSS97] D. A. Spielman, M. A. Shokrollahi, and V. Stemann. A remark on Matrix Rigidity. *Information Processing Letters*, 64(6):283–285, 1997.

[Sud78] I. Sudborough. On the tape complexity of deterministic context-free language. *Journal of Association of Computing Machinery*, 25(3):405–414, 1978.

[SW91] Miklos Santha and Christopher Wilson. Polynomial Size Constant Depth Circuits with a Limited Number of Negations. In *Proceedings of the 8th annual symposium on theoretical aspects of computer science (STACS 91)*, volume 480 of *Lecture Notes in Computer Science*, pages 228–237, 1991.

[Tod91] Seinosuke Toda. Counting problems computationally equivalent to the determinant. Technical Report CSIM 91-07, Dept of Comp Sc & Information Mathematics, Univ of Electro-Communications, Chofu-shi, Tokyo, 1991.

[TT86] R. Tamassia and I. G. Tollis. A unified approach to visibility representations of planar graphs. *Discrete and Computational Geometry*, 1(1):312–341, 1986.

[TT89] R. Tamassia and I. G. Tollis. Tessellation representations of planar graphs. In *Proc. 27th Annual Allerton Conference on Communications, Control and Computing, UIUC*, pages 48–57, 1989.

[Val77] Leslie G. Valiant. Graph Theoretic Arguments in Low-level Complexity. In *Proceedings of 6th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 53 of *LNCS*, pages 162–176. Springer, Berlin, 1977.

[Val92] Leslie G. Valiant. Why is Boolean Complexity Theory Difficult? In *Proceedings of the London Mathematical Society symposium on Boolean function complexity*, pages 84–94, New York, NY, USA, 1992. Cambridge University Press.

[Ven91] H. Venkateswaran. Properties that characterize LogCFL. *Journal of Computer and System Sciences*, 42:380–404, 1991.

[Vin91] V. Vinay. Counting Auxiliary Pushdown Automata and Semi-Unbounded Arithmetic Circuits. In *Proc. 6th Structure in Complexity Theory Conference*, volume 223 of *Lecture Notes in Computer Science*, pages 270–284, Berlin, 1991. Springer.

[Vin05] N. V. Vinodchandran. A note on the circuit complexity of pp. *Theoretical Computer Science*, 347(1-2):415–418, 2005.

[Vol99] H. Vollmer. *Introduction to Circuit Complexity: A Uniform Approach*. Springer New York Inc., 1999.

[Whi73] A. T. White. *Graphs, Groups and Surfaces*. North-Holland, Amsterdam, 1973.

[Woo01] D. R. Wood. Geometric thickness in a grid of linear area. In *Proceedings of European Conference Combinatorics, Graph Theory and Applications*, pages 310–315, 2001.

[Yan91] H. Yang. An NC Algorithm for the General Planar Monotone Circuit Value Problem. In *Proc. 3rd IEEE Symp. on Parallel and Distributed Processing*, pages 196–203, 1991.

[Yao79] Andrew C. Yao. Some complexity questions related to distributive computing(preliminary report). In *Proceedings of the eleventh annual ACM symposium on Theory of computing (STOC)*, pages 209–213, New York, NY, USA, 1979. ACM Press.