Complexity of Matrix Rank and Rigidity

Jayalal Sarma M.N.

Joint work with

Meena Mahajan

IMSc

Institute Seminar Week

Mar 5, 2007

Matrix Rank

Rank of a matrix $M \in \mathbb{F}^{n \times n}$ has the following equivalent definitions.

- The size of the largest submatrix with non-zero determinant.
- The number of linearly independent rows/columns of a matrix.
- The smallest r such that $M = AB$ where $A \in \mathbb{F}^{n \times r}$ is an $B \in \mathbb{F}^{r \times n}$ matrix.
- The smallest k such that M is the sum of k rank-1 matrices.

SINGULAR: Given a matrix M, is $\text{rank}(M) < n$?

RANK BOUND: Given a matrix M and a value r, is $\text{rank}(M) < r$?

Computing the Rank

Some motivation ...

- From Linear Algebra : Computation of the dimension of the solution space of a system of linear equations.
- From Control Theory : Rank of a matrix can be used to determine whether a linear system is controllable, or observable.
- From Algorithmics : Some natural algorithmic problems can be expressed in terms of rank computation and determinant computation.
- From Complexity Theory : In the context of seperating complexity classes, it might facilitate application of the well developed algebraic techniques.

Complexity Theoretic Preliminaries

Complexity Theoretic Preliminaries

Classes based on Turing machine (TM) models

- L : Problems that can be solved by TMs which uses only $O(\log n)$-space.

Complexity Theoretic Preliminaries

Classes based on Turing machine (TM) models

- L : Problems that can be solved by TMs which uses only $O(\log n)$-space.
- NL : Problems that can be solved by $O(\log n)$-space TMs which are allowed to guess and verify but they should get it right atleast once for a positive answer and none otherwise.

Complexity Theoretic Preliminaries

Classes based on Turing machine (TM) models

- L : Problems that can be solved by TMs which uses only $O(\log n)$-space.
- NL : Problems that can be solved by $O(\log n)$-space TMs which are allowed to guess and verify but they should get it right atleast once for a positive answer and none otherwise.
- $C_=L$: Problems that can be solved by $O(\log n)$-space TMs which are allowed to guess and verify but they should get it right exactly half the time for a postive answer, and non-half otherwise.

Complexity Theoretic Preliminaries

Classes based on Turing machine (TM) models

- L : Problems that can be solved by TMs which uses only $O(\log n)$-space.
- NL : Problems that can be solved by $O(\log n)$-space TMs which are allowed to guess and verify but they should get it right atleast once for a positive answer and none otherwise.
- $C_=L$: Problems that can be solved by $O(\log n)$-space TMs which are allowed to guess and verify but they should get it right exactly half the time for a postive answer, and non-half otherwise.

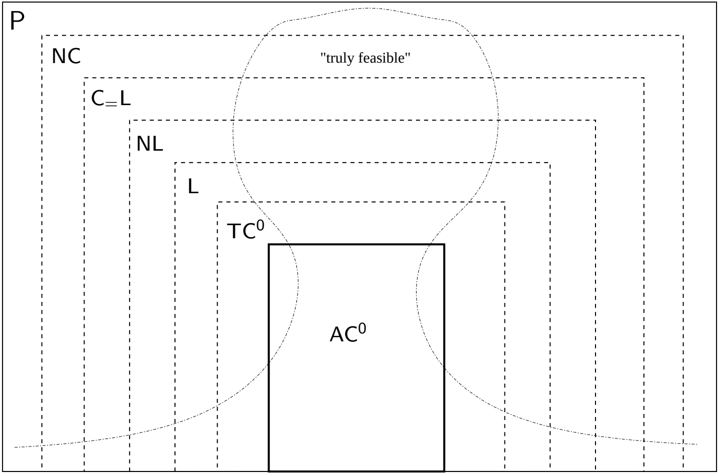
Circuits are directed acyclic graphs with $\wedge$, $\vee$ and $\neg$ gates at the vertices.

- AC^0 : poly size constant depth and unbounded fanin circuits.
- TC^0 : AC^0 with "majority" gates

Why are they interesting ?

Class	Resource Bound	Complete Problem
L	log space TM	Reachability in undirected graphs
NL	log space TMs with guess & verify power	Reachability in directed graphs
$C_=L$	log space TMs with "balanced" guess & verify	Singularity of 0-1 matrices
AC^0	poly size, constant depth circuits	Reachability in constant width maze-graphs
TC^0	AC^0 + "majority"	Testing Majority

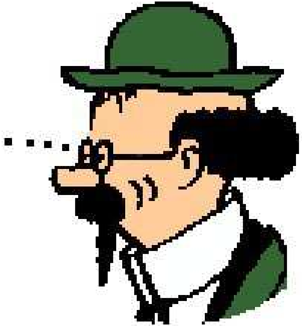
The Zoo below P



So ?

Turing/Circuit Model : Combinatorial !

Seperation of small classes : Unknown



Rank Computation : Algebraic !

Characterising computation using this might help.

Computing the Rank

- The natural approach takes exponential time.
- Can be computed in Polynomial time :
  Gaussian elimination (1800s)
  But it is inherently sequential.
- Elegant parallel algorithm ([Mul87]) by relating the problem to testing if some coefficients of the characterstic polynomial are zeros. Rank can be computed in NC.
- Refined complexity bounds by [ABO96]. Upperbound testing exactly characterises $C_=L$.

Computing the rank of special matrices

- Complexity theoretic characterisations.
- Several applications have inherent structure for the matrices.

Restrictions we are interested in :

- $M = [a_{i,j}]$ is diagonally dominant if

$$|a_{ii}| \geq \sum_{j \neq i} |a_{ij}|$$

  Fun fact : If dominance is strict for all i, M is non-singular.
- Diagonal matrices : Non-zero entries only on the main diagonal.

Characterising Log space

Theorem

Computing the rank of symmetric non-negative diagonally dominant matrices is complete for the complexity class L.

- Membership: The proof uses a nice combinatorial characterisation of the dimension of the null-space of the matrix due to [Dah99]. We can reduce the problem to counting the number of bipartite components in a graph.
- Hardness :The problem of testing reachability in undirected forests where there are exactly two components is L-complete. We reduce this problem to rank computation on symmetric non-negative diagonally dominant matrices.

For Special Matrices...

Matrix type	RANK BOUND	SINGULAR
General	$C_=L$-complete [ABO96]	$C_=L$-complete [ABO96]
Sym.Non-neg.	$C_=L$-complete [ABO96]	$C_=L$-complete [ABO96]

For Special Matrices...

Matrix type	RANK BOUND	SINGULAR
General	$C_=L$-complete [ABO96]	$C_=L$-complete [ABO96]
Sym.Non-neg.	$C_=L$-complete [ABO96]	$C_=L$-complete [ABO96]
Sym.Non-neg. Diag. Dom.	L-complete	L-complete
Diag	TC^0-complete	in AC^0

How close is M to a rank r matrix?

Definition (Rigidity)

Given a matrix M and $r < n$, rigidity of the matrix M ($R_M(r)$) is the number of entries of the matrix that we need to change to bring the rank below r.

- A natural linear algebraic optimisation problem; and it arises in control theory.
- Interesting in a circuit complexity theory setting. Highly rigid linear transformations(matrices) have some "nice" size lowerbounds for log-depth circuits computing them [Val77].
- It is related to the "power" of Valiant's proof technique.

Computing Rigidity

$\textsc{Rigid}(M, r, k)$: Given a matrix M, values r and k, is $R_M(r) \leq k$.

- Over any finite field $\mathbb{F}$, $\textsc{Rigid}$ is in NP. The algorithm will simply guess the positions and the changed values and simply verify if the rank has gone down.
- Over $\mathbb{F}_2$, $\textsc{Rigid}$ is NP-complete [Des]. The hardness comes from a reduction from a problem in the coding theory setting : the nearest neighbour decoding problem.
- Over infinite fields the only upperbound we know is $r.e.$
- If k is constant, restriced to boolean matrices, $\textsc{Rigid}$ is $C_=L$-complete.

Bounded Rigidity

Too much change involves too much "cost".

- Given a matrix M and $r < n$, bounded rigidity of the matrix M ($R_M(b, r)$) is the number of entries of the matrix that we need to change to bring the rank below r, if the change allowed per entry is atmost b.
- $\text{B-RIGID}(M, r, k, b)$: Given a matrix M, values b, r and k, is $R_M(b, r) \leq k$?
- Another formulation : Define an interval of matrices $[A]$ where

$$m_{ij} - b \leq a_{ij} \leq m_{ij} + b$$

Question : Is there a rank r matrix $B \in [A]$ such that $M - B$ has atmost k non-zero entries?

Why should there be?

Consider the matrix

$$\begin{bmatrix} 2^k & 0 & 0 & 0 & 0 \\ 0 & 2^k & 0 & 0 & 0 \\ 0 & 0 & 2^k & 0 & 0 \\ 0 & 0 & 0 & 2^k & 0 \\ 0 & 0 & 0 & 0 & 2^k \end{bmatrix}$$

- $R_M(b, n-1)$ is undefined unless $b \geq \frac{2^k}{n}$.
- Question : For a given matrix M, bound b, target rank r, can we efficiently test, whether $R_M(b, r)$ is defined ?

Why should there be?

Consider the matrix

$$\begin{bmatrix} 2^k & 0 & 0 & 0 & 0 \\ 0 & 2^k & 0 & 0 & 0 \\ 0 & 0 & 2^k & 0 & 0 \\ 0 & 0 & 0 & 2^k & 0 \\ 0 & 0 & 0 & 0 & 2^k \end{bmatrix}$$

- $R_M(b, n-1)$ is undefined unless $b \geq \frac{2^k}{n}$.
- Question : For a given matrix M, bound b, target rank r, can we efficiently test, whether $R_M(b, r)$ is defined ?

It is NP-hard.

A restricted case

For a given matrix M, bound b, testing whether $R_M(b, n-1)$ is defined, is NP-complete.

- Membership: The bound b defines an interval for each entry of the matrix.
  Determinant is a multlilinear polynomial on the entries of the matrix.
  Now use the following lemma:

Lemma (Zero-on-an-edge)

For a multilinear polynomial $p(x_1, x_2 \ldots x_t)$, consider the hypercube defined by the interval of each of the x_is. If there is a zero of the polynomial in the hypercube then there is a zero on an edge of the hypercube.

- NP algorithm : Guess the “nice” singular matrix and verify.
- Hardness: A reduction from MAX-CUT problem.

Thank You

Eric Allender, Robert Beals, and Mitsunori Ogihara.
The complexity of matrix rank and feasible systems of linear equations.
In *STOC 96*, pages 161–167, 1996.

G. Dahl.
A note on nonnegative diagonally dominant matrices.
Linear Algebra and Applications, 317:217–224, April 1999.

Amit Jayant Deshpande.
Private Communication (Sep 2006).

K. Mulmuley.
A fast parallel algorithm to compute the rank of a matrix over an arbitrary field.
Combinatorica, 7:101–104, 1987.

L. G. Valiant.
Graph theoretic arguments in low-level complexity.
In *MFCS 1977*, pages 162–176, 1977.