The Parallel Complexity of Positive Linear Programming

Luca Trevisan*
Univ. di Roma *La Sapienza*
Dipart. di Sci. dell'Informazione
Via Salaria 113, I-00198 Roma, Italy
Email: trevisan@dsi.uniroma1.it

Fatos Xhafa†
Departament de LSI
Facultat d'Informàtica, UPC
Pau Gargallo 5, 08028 Barcelona, Spain
Email: fatos@goliat.upc.es

September 1996

Abstract

In this paper we study the parallel complexity of Positive Linear Programming (PLP), i.e. the special case of Linear Programming in packing/covering form where the input constraint matrix and constraint vector consist entirely of positive entries. We show that the problem of exactly solving PLP is P-complete. Luby and Nisan gave an NC approximation algorithm for PLP, and their algorithm can be used to approximate the size of the largest matching in bipartite graphs, or to approximate the size of the set cover to within a factor $(1+\epsilon)\ln\Delta$, where Δ is the maximum degree in the set system. Trevisan used positive linear programming in combination with Luby and Nisan's algorithm to obtain an NC $(3/4-\epsilon)$-approximate algorithm for Max SAT. An important implication of our result is that, by using the Linear Programming technique, we cannot exactly compute in NC the cardinality of Maximum Matching in bipartite graphs or finding a $(\ln\Delta)$-approximation for Minimum Set Cover, or a 3/4-approximation of an instance of Maximum SAT, unless P=NC.

1 Introduction

Linear Programming (LP) is one of the most central problems in combinatorial optimization. It is the problem of optimizing a linear function $\mathbf{c}^T\mathbf{x}$ over a convex polyhedron $\{\mathbf{x} : A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq 0\}$, where $\mathbf{x} \in \mathsf{R}^n_+$, A is an $m \times n$-matrix and $\mathbf{b}, \mathbf{c} \in \mathsf{R}^n$. The parallel complexity of this problem is, by now, well understood. Dobkin, Lipton, Reiss and Khachyan [4, 8] showed that (the general) LP was complete, in the strong sense, for P under *logspace* reductions. Later on, it was shown that even the problem of approximating the value of a general linear program is also P-complete [13, 12]. Therefore, there is no fast parallel algorithm for solving LP or for approximating it, unless P=NC. However, these results do not exclude out the existence of NC algorithms[1] for special cases of LP. Indeed, Luby and Nisan [11] gave an NC approximation algorithm for the restricted version of linear programming called Positive Linear Programming (PLP). An instance of PLP has all the entries of the matrix A and those of $\mathbf{b}$ and $\mathbf{c}$ non-negative, and it is in the *packing* (resp. *covering*) form, i.e., the linear restrictions are given by $A\mathbf{x} \leq \mathbf{b}$ and the objective function is to be maximized (resp. $A\mathbf{x} \geq \mathbf{b}$, the objective function is to be minimized). Luby and Nisan's algorithm computes a feasible $(1+\epsilon)$-approximate solution in time polynomial in $1/\epsilon$ and $\log N$, using $O(N)$ processors

*Research done while visiting the Departament de Llenguatges i Sistemes Informàtics de la UPC.

†The research of this author was supported by ESPRIT Long Term Research Project 20244-ALCOM-IT.

[1] i.e., algorithms that run in *polylog* time and use a *polynomial* number of processors

(where N is the size of the input). An algorithm with such a trade-off between approximation guarantee and efficiency is usually called an NCAS (NC Approximation Scheme) (see, e.g., [3]).

PLP is of a particular interest since many important combinatorial problems can be casted by positive linear programs and therefore Luby and Nisan algorithm can be used to approximate them in NC. Thus, Maximum Matching in bipartite graphs [5, 6] is modeled by a positive 0/1 linear program, and if we relax the condition for 0/1 variables be simply positive the optimum value is not changed. Therefore, Luby and Nisan's algorithm can be used to approximate the size of a largest matching, and as indicated in [11], this is essentially the result of [2]. Also, Minimum Set Cover can be formulated as a 0/1 positive linear program [10]. In this case, relaxing the condition for the integrality of variables decreases the optimum by a factor of $\ln \Delta$, where Δ is the maximum degree in the set system. Therefore, the algorithm for PPL approximates the optimum size of the set cover within a factor of $(1+\epsilon)\ln \Delta$. The use of PLP in the design of parallel approximation algorithms has been further explored in [14]. Among other results, a PLP relaxation of Maximum Satisfiability (Max SAT) is presented whose optimum is at most 3/4 times the optimum of the Max SAT problem. In combination with Luby and Nisans algorithm this gives an NC $(3/4-\epsilon)$-approximate algorithm for Max SAT.

Unfortunately, Luby and Nisan's algorithm cannot be used to exactly solve an instance of PLP in NC. In this note we address the problem of the parallel complexity of PLP. We show that the problem of exactly solving PLP is P-complete. Our result is based on the observation that the *Circuit Value Problem* (CVP), which is P-complete [9], can be logspace reduced to PLP. The reduction follows that of [7] but we take care of the linear constraints and the objective function to have non-negative coefficients. An important implication of our result is that, by using the LP technique, we cannot exactly compute in NC the cardinality of Maximum Matching in bipartite graphs or finding a $(\ln \Delta)$-approximation for Minimum Set Cover, or a 3/4-approximation of an instance of Maximum SAT, unless P=NC.

Preliminaries

An instance of CVP is: "Given an encoding of a Boolean circuit that consists of computational gates `NOT` and `OR`[2] together with an input assignment, determine whether the output gate evaluates to 0 or 1." We assume that the reader is familiar with the notion of logspace reductions, and is referred to [1, 7] for definitions. We denote by $(A, \mathbf{b}, \mathbf{c})$ an instance of PLP in the packing form. The corresponding decision version of this problem is: "Given an instance $(A, \mathbf{b}, \mathbf{c})$ and $d \in \mathsf{R}^+$, is there any vector $\mathbf{x} \in \mathsf{R}^n_+$, such that $A\mathbf{x} \leq \mathbf{b}$ and $\mathbf{c}^T\mathbf{x} \geq d$?" We will denote by $(A, \mathbf{b}, \mathbf{c}, d)$ an instance of this problem. We will use boldface character (e.g. $\mathbf{t}$) to denote vectors; sometimes we will use $\mathbf{1}$ to denote a vector all whose entries are equal to 1. Finally, for a set I, we denote by $|I|$ its cardinality.

2 The P-completeness of Fractional Packing Problems

We recall the standard reduction from the CVP to Linear Programming. Let $g_1, \ldots, g_m$ be the gates of the circuit, we use a variable t_i for any gate g_i. The intended meaning of such variables will be that $t_i \in \{0, 1\}$ and that $t_i = 1$ iff the output of g_i is one. We associate one or more linear constraints to any gate: the constraints will be such that only one feasible solution exists (namely, the solution in which the values of t_i are consistent with their intended meaning). If g_k

[2]This version of CVP has also been shown to be P-complete.

is an input gate whose value is zero (resp. one), then the corresponding constraint will be $t_k = 0$ (resp. $t_k = 1$). If g_k is a `NOT` gate whose input comes from gate g_j, then the constraint will be $t_k = 1 - t_j$. Finally, if g_k is an `OR` gate whose inputs come from gate g_i and g_j, then the constraints will be $t_k \geq t_i$, $t_k \geq t_j$, $t_k \leq t_i + t_j$. For all the gate variables we also have $0 \leq x_i \leq 1$ [7]. It is easy to prove by induction on the depth of the circuit that such linear program has only one feasible solution, namely the solution that corresponds to the correct settings of the gates. Thus, if we use t_m as objective function, the optimum value will be zero or one, and will be one iff the circuit outputs one.

The above described linear program can be expressed as

$$
\begin{array}{lll}
\max & t_m & \\
\text{subject to} & & \\
 & t_k = 1 & \forall k \in In1 \\
 & t_k = 0 & \forall k \in In0 \\
 & & \\
 & t_k \geq t_i & \forall (i,j,k) \in OR \\
 & t_k \geq t_j & \forall (i,j,k) \in OR \\
 & t_k \leq t_i + t_j & \forall (i,j,k) \in OR \\
 & & \\
 & t_k = 1 - t_j & \forall (j,k) \in Neg \\
 & & \\
 & 0 \leq t_i \leq 1 & \forall i \in \{1, \ldots, m\}
\end{array}
\tag{LP1}
$$

where we used the notation $In0$ (resp. $In1$) to denote the set of indices of input gates whose value is zero (resp. one), the notation Neg to denote the set of pair of indices (j,k) such that g_k is a `NOT` gate taking its input from g_j, and OR to denote the set of triples (i,j,k) such that g_k is an `OR` gate taking its inputs from gates g_i and g_j.

Clearly, the program (LP1) is not an instance of PLP. Notice that in (LP1) we have some constraints which are equalities and also there are variables with negative coefficients. We will deal with both of them in two separate steps. We first introduce new variables $f_1, \ldots, f_m$ such that $f_i = 1 - t_i$. The program becomes

$$
\begin{array}{lll}
\max & t_m & \\
\text{subject to} & & \\
 & t_k = 1 & \forall k \in In1 \\
 & f_k = 1 & \forall k \in In0 \\
 & & \\
 & f_k + t_i \leq 1 & \forall (i,j,k) \in OR \\
 & f_k + t_j \leq 1 & \forall (i,j,k) \in OR \\
 & f_i + f_j + t_k \leq 2 & \\
forall(i,j,k) \in OR & & \\
 & & \\
 & t_k + t_j = 1 & \forall (j,k) \in Neg \\
 & & \\
 & t_i + f_i = 1 & \forall i \in \{1, \ldots, m\} \\
 & t_i, f_i \geq 0 & \forall i \in \{1, \ldots, m\}
\end{array}
\tag{LP2}
$$

It should be clear that there is a correspondence between the unique feasible solution of (LP1) and the unique feasible solution of (LP2), more formally, we have the following result.

Fact 1 *If* $\mathbf{t}$ *is a feasible solution for (LP1), then* $(\mathbf{t}, \mathbf{1} - \mathbf{t})$ *is a feasible solution for (LP2), and the cost of the solutions are equal. If* $(\mathbf{t}, \mathbf{f})$ *is a feasible solution for (LP2), then* $\mathbf{t}$ *is a feasible solution for (LP1) and the cost of the solutions are equal.*

Note that (LP1) is not yet a packing problem, since there are equality constraints. The final step will be to relax them into inequality constraints and to modify the objective function in such a way that it will never be "convenient" to strictly satisfy the relaxed constraints. We note that our technique bears some similarity to the method of Lagrangean relaxations.

$$
\begin{array}{lll}
\max & t_m + \sum_{k \in In1} t_k + \sum_{k \in In0} f_k + \sum_{(k,j) \in Neg} (t_k + t_j) + \sum_{i=1}^{m} (t_i + f_i) & \\
\text{subject to} & & \\
& t_k \leq 1 & \forall k \in In1 \\
& f_k \leq 1 & \forall k \in In0 \\
& & \\
& f_k + t_i \leq 1 & \forall (i,j,k) \in OR \\
& f_k + t_j \leq 1 & \\
& f_i + f_j + t_k \leq 2 & \\
& & \\
& t_k + t_j \leq 1 & \forall (j,k) \in Neg \\
& & \\
& t_i + f_i \leq 1 & \forall i \in \{1, \ldots, m\} \\
& t_i, f_i \geq 0 & \forall i \in \{1, \ldots, m\}
\end{array}
\qquad \text{(LP3)}
$$

Lemma 1 *There exists a solution for (LP3) of cost* $1 + |In1| + |In0| + |Neg| + m$ *if and only if there exists a solution for (LP2) of cost 1.*

Proof: Let $K = 1 + |In1| + |In0| + |Neg| + m$. It is immediate to verify that a feasible solution for (LP2) of cost 1 is feasible for (LP3) and its cost is K. Assume now that $(\mathbf{t}, \mathbf{f})$ is feasible for (LP3) and its cost is K; we claim that $(\mathbf{t}, \mathbf{f})$ is feasible for (LP2) and that $t_m = 1$. Indeed, the cost of a solution for (LP3) is the sum of K terms, and each one is constrained to be at most one. If there exists a feasible solution whose cost is K, then it follows that all such terms are equal to one, and thus the solution is feasible for (LP2) and $t_m = 1$. □

Remark 2 *One would be tempted to conjecture the following stronger statement of Lemma 1:* If we let Z_2^* (resp. Z_3^*) be the optimum solution of (LP2) (resp. (LP3)), then $Z_3^* = Z_2^* + |In1| + |In0| + |Neg| + m$. *Unfortunately, there are counterexamples of such statement.*

Theorem 3 *PLP in packing form is P-complete.*

Proof: It is immediate to check that, given the description of a circuit, the PLP instance (LP3) can be constructed using logarithmic space. The theorem thus follows from Lemma 1. □

The P-hardness of optimally solving PLP covering problems immediately follows from the duality theorem of linear programming (covering problems are the duals of packing problems). The P-completeness of the decision version can be established directly by minor changes to the above proof.

3 Conclusions

Our result shows that Luby and Nisan's algorithm cannot be improved to the point of computing optimum solutions in NC for fractional packing and covering problems. However, it is still not clear whether there exists a fully NC approximation scheme (FNCAS) for such problems (recall that a FNCAS is an NCAS whose running time is polynomial in $\log 1/\epsilon$ and N, where N is the size of the input). Remark 2 shows that our reduction is not sufficient to rule out this possibility.

References

[1] J.L. Balcázar, J. Díaz, and J. Gabarró. *Structural Complexity I.* Springer Verlag, 1995.

[2] E. Cohen. Approximate maxflow on small depth networks. In *Proceedings 37th IEEE Symposium on Foundations of Computing Science*, pages 648–658, 1992.

[3] J. Díaz, M.J. Serna, P. Spirakis, and J. Torán. *Paradigms for fast parallel approximations.* Cambridge University Press (To appear).

[4] D. Dobkin, J.R. Lipton, and S. Reiss. Linear Programming is logspace hard for P. *Information Processing Letters*, 8:96–97, 1979.

[5] J. Edmonds. Maximum matching and a polyhedron with 0,1-vertices. *Journal of Research of the National Bureau of Standards B*, 69B:125–130, 1965.

[6] J. Edmonds and E. Johnson. Matching: A well-solved class of integer linear programs. In *Proceedings of the Calgary International Conference on Combinatorial Structures and their Applications*, pages 82–92, 1970.

[7] R. Greenlaw, H.J. Hoover, and W.L. Ruzzo. *Limits to parallel computation: P–completeness theory.* Oxford University Press, 1995.

[8] L. G. Khachyan. A polynomial algorithm in linear programming. *Translated in Soviet Mathematics Doklady*, 20:191–194, 1979.

[9] R.E. Ladner. The circuit value problem is logspace complete for P. *SIGACT News*, 7:18–20, 1975.

[10] L. Lovász. On the ratio of optimal integral and fractional covers. *Discrete Mathematics*, 13:383–390, 1975.

[11] M. Luby and N. Nisan. A Parallel Approximation Algorithm for Positive Linear Programming. In *Proceedings 25th ACM Symposium on Theory of Computing*, pages 448–457, 1993.

[12] N. Megiddo. A note on approximate linear programming. *Information Processing Letters*, 42:53, 1992.

[13] M.J. Serna. Approximating linear programming is log-space complete for P. *Information Processing Letters*, 37:233–236, 1991.

[14] L. Trevisan. Positive Linear Programming, Parallel Approximation and PCP's. In *Fourth European Symposium on Algorithms*, volume 1136 of *Lecture Notes in Computer Science*, pages 62–75. Eds. Díaz, J. and Serna, M., Springer-Verlag, 1996.