ON THE COMPUTATIONAL COMPLEXITY OF INTEGER PROGRAMMING PROBLEMS

by

Ravindran Kannan and Clyde L. Monma

University of Bonn, West Germany and Cornell University

The authors were partially supported by N.S.F. Grant ENG-76-09936 and by SFB 21 (DFG), Institut für Operations Research, Universität Bonn, Bonn

Abstract: Recently much effort has been devoted to determining the computational complexity for a variety of integer programming problems. In this paper a general integer programming problem is shown to be NP-complete; the proof given for this result uses only elementary linear algebra. Complexity results are also summarized for several particularizations of this general problem, including knapsack problems, problems which relax integrality or non-negativity restrictions and integral optimization problems with a fixed number of variables.

1. Introduction

Recently much effort has been devoted to determining the computational complexity for a variety of integer programming problems; see [7], [11] and [13] - [19]. This paper examines a general integer programming problem and several of its variations in an attempt to highlight the computational difficulties inherent in integer programming. A basic familiarity with the terms P, NP and NP-Complete is assumed. The interested reader is referred to [1], [6], [9], [16] or [17] for a detailed discussion of these concepts.

The general integer programming question considered, denoted by IP, is: "Given positive integers m and n, an mxn integral

R. Henn et al. (eds.), *Optimization and Operations Research*

matrix A, and an mx1 integral vector b, does there exist an integral vector x satisfying $Ax \geq b$?" Problems will be formulated as feasibility questions with yes or no answers. These questions will be generic with the variable quantities to be specified appearing in the first part and the property to be tested for in the second part. The statement and proof [14] of the known result, see [3], [7] and [11], that IP is in NP requires only a knowledge of elementary linear algebra.

The computational complexity of several variations of IP is also surveyed. These variations include knapsack problems, a group problem, problems which relax integrality or nonnegativity restrictions and integral optimization problems with a fixed number of variables.

Section 2 An elementary proof that IP is in NP.

By the Integer Programming Problem we mean the question: "Given positive integers m and n, an mxn integral matrix A and an mx1 integral vector b, does there exist an integral vector x satisfying $Ax \geq b$?". Let Z^n denote the integer-valued vectors of $\mathbb{R}^n$ and define

(1) $\quad S = \{x : Ax \geq b;\ x \in Z^n\}$.

First, we note that for $S' = \{x : Ax \geq b;\ x \text{ a } (0,1)\text{-vector}\}$, the question "Is S' non-empty?" is certainly in NP because one could nondeterministically write down the values - 0's and 1's - of each x_i, and then check in polynomial time that this x satisfies $Ax \geq b$. This procedure works because the only values of the x_i's to be tested are 0 and 1 (the number of possible values is 2). For the general case, however, if we wish to consider values $x_i = 0,1,2,\ldots,k$, then $\log_2 k$ binary digits are required

to represent x_i. Thus, to prove that the general problem "Is S nonempty?" is in NP, it suffices to prove that there is a fixed polynomial $q(n,m)$ such that the above question can be answered by considering at most $2^{q(n,m)}$ consecutive values of each x_i. This is what we prove below. Let a be the largest absolute value of any entry in A.

Theorem 1: There exists a polynomial $q(.,.)$ such that for any positive integers m and n and any mxn integral matrix A and any mx1 integral vector b, $S = \{x : Ax \geq b;\ x \in Z^n\}$ is nonempty if and only if there is an x in S such that $||x||$, the maximum absolute value of any entry of x, is at most $2^{q(n,m)}$.

While J. Gathen and M. Sieveking [11] give a proof of Theorem 1, the proof presented here is shorter and more elementary. Cook [7] and I. Borosh and L.B. Treybig [3] also proved this theorem independently of our work. All four proofs are quite different.

Proof of Theorem 1:

We first prove a preliminary lemma.

Lemma 1: If D is an integral kxn matrix with rank $D < n$, there is an integer vector $f = (f_1, f_2, \ldots, f_n)$ such that:

(i) not all f_i are zero;

(ii) $Df = 0$;

(iii) $||f|| \leq n(dk)^k$, where d is the largest absolute value of any entry of D.

Proof: Without loss of generality, assume that all the rows of D are independent. (If not, choose a basis for the row space of D and delete the other rows.) Now, renumber the columns of D, if necessary, so that the first k columns form a nonsingular matrix

B. Set $f_{k+1} = f_{k+2} = \ldots = f_n = -1$. Let $(f_1, f_2, \ldots, f_k) = f'$ and for $i = 1,2,\ldots,k$, let $\sum_{j=k+1}^{n} d_{ij} = \lambda_i$. Since B is nonsingular, there is a solution to $Bf' = \lambda$, where $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_k)^T$. Note that then $(f', f_{k+1}, f_{k+2}, \ldots, f_n)$ satisfies $Df = 0$. By Cramer's rule, f' is a rational vector. Further, for each i, $1 \leq i \leq k$, f'_i has a numerator which is a (kxk) determinant with one column given by λ and the rest columns of B. Since $\|\lambda\| \leq nd$ and each entry of B is at most d in absolute value, $|\text{numerator}| \leq ((nd)\, d^{k-1})k!$ which is at most $n(dk)^k$. (f·det B) provides the necessary solution satisfying (i), (ii) and (iii) of the lemma.

Now, S defined in (1) is nonempty if and only if $S \cap \{x : I_i x_i \geq 0,\ i = 1,2,\ldots,n\}$ is nonempty for some choice of $\{I_i\}_{i=1}^{n}$ each I_i being choosen from the two element set $\{-1,1\}$. Thus, it is enough to prove that in each of these 2^n sets, there exists an x if an only if there exists one satisfying $\|x\| \leq 2^{q(n,m)}$ ($q(n,m)$ will be computed later). Note that the addition of the n inequalities makes the rank of the coefficient matrix equal to n. Thus, without loss of generality, we may assume that rank $A=n$ and prove the theorem for S itself. The addition of the n constraints increases the number of rows of A by n, thus A will be assumed to be an Mxn matrix where $M=m+n$.

In the rest of the proof, we will assume as hypothesis that S is nonempty. There is an x in S such that $a^i \cdot x \leq b_i + a$ for at least one $i \in \{1,2,\ldots,M\}$, where a^i stands for the ith row of A. (To see this, take any $x \in S$ and alter a component of x until the above inequality is satisfied.) Without loss of generality assume $i=1$. (If not, renumber the rows of A.) Therefore, there is an integer b'_1 with $b_1 \leq b'_1 \leq b_1 + a$ for which

$$S_1 = S \cap \{x \mid a^1 \cdot x = b'_1\} \neq \emptyset.$$

For induction, assume that there are positive integers $b'_1, b'_2, \ldots, b'_k$ such that b'_i is at most $b_i + n^2 a (aM)^M$ for $i=1,2,..,k$. (It will become clear later why this bound is chosen.) and

$$S_k = S \cap \{x \mid a^i \cdot x = b'_i \text{ for } i=1,2,\ldots,k\} \neq \emptyset.$$

Let A_k denote the first k rows of A. There are two cases to consider.

Case 1: rank $A_k < n$:

By Lemma 1, there is a $f \neq 0$, such that $A_k f = 0$ with f integral and $\|f\| \leq n (aM)^M$. If x solves $A_k x = (b'_1, b'_2, \ldots, b'_k)^T$ then $(x + cf)$ solves the same k equations for any integer c. Using this fact, we prove:

Lemma 2: $S_k \neq \emptyset$ implies that there is an x in S_k such that

(*) $a^i x \leq b_i + na\|f\|$ for some i, $k+1 \leq i \leq M$.

Proof: Suppose there is an x in S_k (recall $S_k \neq \emptyset$) not satisfying (*). We will show that we can add a suitable multiple of f to x to get an $(x + cf) \in S$ satisfying (*).

Since A is of rank n, $a^{i_o} f \neq 0$ for some $i_o \geq k+1$. Replacing f by $-f$, if necessary, we can assume that $a^{i_o} \cdot f < 0$.

Let

$$\alpha = \min_{\{i:\ a^i \cdot f < 0\}} \left\lfloor \frac{a^i x - b_i}{|a^i \cdot f|} \right\rfloor \qquad (2)$$

where $\lfloor z \rfloor$ is the largest integer less than or equal to z.

Then it is easily checked that $(x+\alpha f) \in S_k$. If i_1 is the value

of i for which the minimum is achieved in (2), then a simple calculation shows that $a^{i_1}(x + \alpha f) \leq b_{i_1} + |a^{i_1} \cdot f|$ which is at most $b_{i_1} + na\ ||f||$ since $|a^i \cdot f| \leq na\ ||f||$ for any i. Hence the lemma follows.

Without loss of generality, assume that the i_1 in the proof of Lemma 2 is $(k+1)$. Thus, there is a b'_{k+1} with $b_{k+1} \leq b'_{k+1} \leq b_{k+1} + na\ ||f||$ such that

$$S_{k+1} = S \cap \{x | a^i \cdot x = b'_i;\ i=1,2,\ldots,k+1\} \neq \emptyset.$$

Since $na\ ||f|| \leq n^2 a (aM)^M$, we have proved the inductive step. Since there are finitely many inequalities and since rank $A = n$, we must finally end in the case where rank $A_k = n$.

Case 2: rank $A_k = n$.

There is now at most one rational x satisfying $A_k \cdot x = (b'_1, \ldots, b'_k)^T$ and by Cramer's rule, the numerators of the x_i in this x are of magnitude at most

$$[||b|| + n^2 a\ (aM)^M]\ a^{M-1}\ M!$$
$$\leq [||b|| + n^2 a (aM)^M]\ (aM)^M.$$

Thus with the polynomial $q(n,m) = \log_2 ([\,||b|| + n^2 a (aM)^M]\ (aM)^M)$ Theorem 1 is true. Note that q is also a polynomial in log a and log $||b||$. Thus, q is a polynomial function of the length of the input data.

3. Variations of IP. The computational complexity of IP and several of its variations is summarized in Table I. The first part of the table contains feasibility problems. The general form is: "Given the problem data, does there exist an x satisfying the generic property?" The second part of the table contains optimi-

zation problems of the form: "Given the problem data, find an x, if one exists, which maximizes cx subject to the generic property." All of the problem data is assumed to be integer-valued. Other restrictions on the input values are indicated by enclosing the parameter and its restriction in parenthesis; e.g., $(m=1)$ means that m is fixed at the value of one. Throughout A is an mxn matrix. The remainder of this section will use these results to highlight some of the computational difficulties inherent in integer programming. IP is easily seen to be equivalent to Problem 1 and both formulations will be referred to as IP.

One might conjecture that the wide range of possible values which A, b, and x may assume results in the computational difficulties for IP. However, the proof that Problem 2 is NP-Complete, see [16], reveals that this fact remains true even when the entries in A are 0,1 or -1, the entries in b are 1, 0, -1 or -2, each row of A has at most three nonzero entries and x is restricted to be a (0,1)-valued vector. Hence, $n \leq 3m$. This indicates that the computational difficulties for IP arise from some underlying structure which is independent of large problem data.

The interplay between the integrality and nonnegativity restrictions of IP will be examined next. Problem 3 relaxes non-negativity restrictions on variables x_B which correspond to a nonsingular submatrix B of A. The system $\{Bx_B + Nx_N = b;$ (x_B, x_N) integer; $x_N \geq 0\}$ is equivalent to $\{B^{-1}Nx_N = B^{-1}b(\text{mod } 1);$ $x_N \geq 0$; integer$\}$. Each equations can be multiplied by $d = |\det B|$, to obtain the equivalent system $\{A'x_N = b'(\text{mod } d);\ x_N \geq 0,$ integer$\}$, where A' and b' are integer valued. This is an instance of Problem 4. Problem 4 will be shown to be in P which will imply

that Problem 3 is in P. Multiplying each congruence i by $(\pi_{j \neq i} d_j)$ yields an equivalent system {A'x = b'(mod d); x $\geq$ 0, integer}. The polynomial algorithm to reduce A' to a diagonal matrix A" , given in [5], can be applied to this system using arithmetic (mod d) to obtain an equivalent system {A"x = b" (mod d); x $\geq$ 0, integer}, where A" has integer diagonal elements $a"_{11}$, $a"_{22}$, ..., $a"_{mm}$ and b" is integer valued. The problem has reduced to checking if there exists an x satisfying $a"_{ii} x_i = b"_i$ (mod d); $x_i \geq 0$, integer for $1 \leq i \leq m$. Each of these m problems is in P. Relaxing all of the nonnegativity restrictions of IP yields the integer feasibility Problem 5, denoted by IF, which is in P [20]. Relaxing all of the integrality restrictions of IP yields the linear programming Problem 6, denoted by LP, whose computational complexity is a major open question. (We thank A. Bachem [2] for his helpful discussions of Problems 3-5.)

It is of interest that both LP and IF may be stated as theorems of the alternative. For LP this yields the well-known Farkas Lemma:

Either (a) there exists a nonnegative vector x
such that Ax = b, or
(b) there exists a y such that yA is
nonnegative and yb is negative,

but not both.

For IF this yields the following:

Either (a) there exists an integer vector x
such that Ax = b, or
(b) there exists a y such that yA is
integer and yb is fractional,

but not both.

These differ only in that "nonnegative" is interchanged with "integer." The latter is stated in [8] as a "classical" result.

The effect of fixing the number of constraints, m, is shown by Problems 7-10. Problems with m=1are called knapsack problems. Fixing m in LP results in Problem 7 which is in P. This is evident since only the $O(n^m)$ possible basic feasible solutions need to be checked; see [10; Chapter 2]. However, the other problems remain NP-Complete.

Problems 11 and 12 are optimization problems with the number of variables, n, fixed at the value of two. These are both in P. The computational complexity of these problems for other fixed values of n is an open question.

Finally, the optimization Problem 11 with n allowed to be a variable part of the input is examined. Denote this knapsack optimization problem by KSO. The best known algorithms for KSO require O(nb) time in the worst case; see [12]. These algorithms solve KSO parametrically for all right-hand sides with value no greater than b. An interesting property of KSO is that for any a and c, there exists a b* such that if KSO can be solved efficiently for each right-hand side with value no greater than b* then KSO can be solved efficiently for all right-hand side values. Two upper bounds on b* are available; see [10; Chapter 6]: Suppose that $c_1/a_1 \geq c_2/a_2 \geq \cdots \geq c_n/a_n$, then

(a) $b^* \leq c_1 / (c_1/a_1 - c_2/a_2)$, and

(b) $b^* \leq a_1 (1+a_k) - a_k$, where $a_k = \max\{a_i : 1 \leq i \leq n\}$.

Thus, the computational complexity of KSO arises from solving for right-hand sideswith value no greater than b* and determining b*. The algorithm for KSO can be used to determine b* in

$O(nb^*)$ time; see [12].

Problem Data	Generic Problem	Computational Complexity
Feasibility Problems	Does there exist an x satisfying:	
1. n,m,A,b	$Ax=b$; $x \geq 0$, integer.	NP-Complete [14],[3],[7], [11]
2. n,m,A,b	$Ax \geq b$; $x \in \{0,1\}^n$.	NP-Complete [16]
3. n,m,A=[B,N],b (rank B=m)	$Bx_B + Nx_N = b$; $x=(x_B,x_N)$ integer; $x_N \geq 0$.	P
4. n,m,A,b,(d>0)	$a^i x \equiv b_i$ (modulo d_i), $1 \leq i \leq m$; $x \geq 0$, integer.	P
5. n,m,A,b	$Ax = b$; x integer	P [20]
6. n,m,A,b	$Ax = b$; $x \geq 0$	Open Problem
7. n,(m is fixed) A,b	$Ax = b$; $x \geq 0$.	P
8. n,(m=1), (A≥0), (b≥0)	$Ax = b$; $x \geq 0$, integer.	NP-Complete [18]
9. n,(m=1), (A≥0)	$Ax=b$; $x \in \{0,1\}^n$; $b = \sum_{i=1}^{n} a_i/2$.	NP-Complete [16]
10. n,(m=2), (A≥0),(b≥0)	$a'x \geq b_1$, $a^2x \leq b_2$; $x \geq 0$, integer.	NP-Complete [19]
Optimization Problems	Find an x, if one exists, which maximizes cx subject to:	
11. (n=2),(m=1), (c≥0),(A≥0), (b≥0)	$Ax \leq b$; $x \geq 0$, integer.	P [13], [15]
12. (n=2),m,(c≥0), (A≥0),(b≥0)	$Ax \leq b$; $x \geq 0$, integer.	P [15]

Table I.

Acknowledgement.

Special thanks are due to Professor Leslie E. Trotter, Jr. for his many hours of help and encouragement.

References

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

[2] A. Bachem, personal communication.

[3] I. Borosh, L.B. Treybig, Bounds on Positive Solutions of Linear Diophantine Equations, Proc. Amer. Math. Soc., Vol. 55, 299, 1976.

[4] G.H. Bradley, Algorithm and Bound for the Greatest Common Divisor of n Integer, Comm. ACM 13, 433-436, 1970.

[5] G.H. Bradley, Algorithms for Hermite and Smith Normal Matrices and Linear Diophantine Equations, Mathematics Computation 25, No. 116, 897-907, 1971.

[6] S.A. Cook, The complexity of theorem proving procedures, Proc. Third ACM Symp. on Th. of Computation (1971), 151-158.

[7] S.A. Cook, A short proof that the linear diophantine problem is in NP, Oct., 1976, Unpublished.

[8] J. Edmonds, F.R. Giles, A Min-Max Relation for Submodular Functions on Graphs, Annals of Discrete Mathematics 1, 185-204, 1977.

[9] M.R. Garey, D.S. Johnson, Computers and Intractibility: A Guide to the Theory of NP-Completeness, to appear W.H. Freeman, publisher, 1978.

[10] R.S. Garfinkel, G.L. Nemhauser, Integer Programming, John Wiley, 1972.

[11] J. Gathen, M. Sieveking, Linear integer inequalities are NP-Complete, submitted to SIAM J. Computing.

[12] P.C. Gilmore, R.E. Gomory, The Theory and Computation of Knapsack Functions, Operations Research 14, (1966), 1045-1074.

[13] D.S. Hirschberg, C.K. Wong, A polynomial time algorithm for the knapsack problem with two variables, JACM 23 (1976), 147-154.

[14] R. Kannan, A proof that integer programming is in NP, unpublished, (1976).

[15] R. Kannan, A Polynomial Algorithm For the Two-Variable Integer Programming Problem, Tech. Report 348, Dept. Oper. Res., Cornell Univ., July 1977.

[16] R.M. Karp, Reducibilities among combinatorial problems, in Complexity of Computer Computations, (eds. R.E. Miller, J.W. Thatcher), Plenum Press, (1972), 85-103.

[17] R.M. Karp, On the computational complexity of combinatorial problems, Networks 5, (1975), 44-68.

[18] G.S. Lueker, Two polynomial complete problems in nonnegative integer programming, TR-178, Dept. Computer Science, Princeton Univ., March 1975.

[19] S. Sahni, Computationally related problems, SIAM J. Cpt. 3 (1974).

[20] E. Specker, V. Strassen, Komplexitaet von Entscheidungsproblemen, Chapter IV by J.v.z. Gathen, M. Sieveking, Lecture Notes in Computer Science 43, Springer-Verlag, New York, 1976.