Polynomial Time Linear programs for hard constraint satisfaction problems

Abstract.

A recent result shows that a LP model with 0/1 values is of polynomial complexity.
The paper reports a model for some important NP hard problems, such as the Propositional Satisfiability Problem, the Traveling Salesperson Problem, and the Minimal Set Covering Problem, by means of only two types of constraints: 'choice constraints'and 'exclusion constraints'.
The article presents a linear 0/1 Simplex for solving the obtained integer program. This algorithm always finds a 0-1 integer solution that corresponds to a solution of the Constraint Satisfaction Problem and vice versa.
The paper presents the results of experiments for solving a Conjunctive Normal Form hard cases by linear programming in polynomial time, confirming in practice the polynomial Acceleration of the Simplex SAT solver by means of intelligent pivot selection through neural networks is also decribed.

Keywords:
Discrete Optimization, Linear programming, Simplex,Neural Networks, 0/1 Polytopes, Traveling Salesman.

1. Introduction.

The Boolean satisfiabilty problem (SAT) is the problem of determining if there exists an interpretation that satisfies a Boolean formula, it asks wheter the variables can receive True or False in such a way the formula evaluates to True.
See [Coo71], [GaJ79], [Mac92].

As shown by [MaF92], the propositional satisfiability problem (SAT) plays a crucial role in the real world and in the field of Artificial Intelligence and Operations Research: it is the core of FCSPs.

We describe a novel approach to m-CNF-SAT and to other FCSPs: LP Sover with (0/1) Matrix.

This is a strongly polynomial algorithms. See [Tar86] and [Sch2021]. A first presentation of our approach can be found in [ZiM97]. See [Mon98, 2011, 2013, 2015] too.

1.1. Related work

Modern optimization began with George Dantzig's development of the Simplex algorithm (1947) for Linear Programming.
However, the worst case complexity of the Simplex algorithm is exponential, even if the Simplex typically requires a low-order polynomial number of steps to compute an optimal solution.
Recently, Khachian's Ellipsoid algorithm [Kha79] and Karmarkar's Projective Scaling Algorithm [Kar84], have been introduced that are provably polynomial.

Gael Glorian, Jean-Marie Lagniez, Valentin Montmiral, and Nicolas Szczepaski in [ScS19] propose and evaluate a new CNF encoding based on chromatic number of a graph. Graph colouring is the problem of assigning a minimum number of colors to all verices of a graph such that no adjacent vertices receive the same color.

Keum-Bae Cho (Academia.org) gives an analysis for SAT problems and a classification: random SAT almost esay, Horn-SAT solvable in polynomial time; k-SAt with k>2 is NP complete. The well known programming language called Prolog (Programming in Logic) is based on Horn clauses.

As an example of Horn clause, consider:

(a + b + C).(b + c + D).(A + c + d)

where + means OR the dot means AND, lower case negated literal, upper case nom-nagated.

Execution of a Prolog program is initiated by the user's posting of a single goal, called the query. Logically, the Prolog engine tries to find aresolution / refutation of the negated query. The resolution method used by Prolog is called SLD (Selected Linear resolution withDefinite clauses) . If the negated query can be refuted, it follows that the query, with the appropriate variable bindings in place, is a logical consequence of the program. In that case, all generated variable bindings are reported to the user, and the query is said to have succeeded.
SLD proceedes as follows:

.Given a goal clause, represented as the negation of a problem to be resolved:

~L1 + … + Ln

and an input definite clause

L + …

whose positive literal unifies with Li, SLD derives another goal clause, in which the selected literal is replaced by the negative literlas of the input clause. Resolution can be restricted to a linear sequence of clauses:

C1,C2,...,Ci.

Prolog power derives from the unification mechanism, which combines matvhing of variables and instantiation.

For Constraint Satisfaction in Prolog see [Mon86] a seminal paper which describes an algorithm for a NP hard problem, Timetabling, implemented in Prolog language. The paper had a great impact in the Logic Programming community.
For efficient logic constraint satisfaction consult [Mon89].

Angione C., A. Occhipinti, G. Stracquadanio, G. Nicosia [AOSN, arXiv 1304.0810] present a statistical physics based characterization of the satisfiability proble. They quote our approach of [ZiM97].

[AOSN] describe an algorithm that produces graphs tarting from SAT and analyzes wheter Bose-Einstein condensation occurs. SAT instances follow Bose statistics and winner-takes-all as the ratio of clauses to variables decreases. We also noted the same for our LP solver.Finally, [AOSN] employ

fitness based classification to enhance SAT solvers such as ChainSAT. Chain approach to SAT constructs a chain of variables where each variable is dependent on a previou one. First, SAT problem is traslated to a graph where the vertices are the clauses and the edges the relation between two clauses. This is smilar to our representation.
Wahid Chrabakh and Rich Wolski, University of California Santa Barbara (Academia.edu) present , parallel SAT solver with intelligent backtracking , scheduling sharing of leaned clauses and clause reduction.

For the approach illustrated in the present paper, see also [Mon92], [Mon92b] and [Mon2023], [ZiM97].

Our LP problem has a Matrix (and b e c vectors) with 0/1 values. This has strongly polynomial algorithms. See [Sch2021], [Tar86].

We summarize the Tardos' method:

Th. There exists an algorithm which soles a given rational LP in polynomial space

Th. There exists a strongly polynomial time agorithm for LP with 0-1 conytaint matrix.

2. The Satisfaction of a Conjunctive Normal Form (m-CNF-SAT)

We report here in italics from [ZiM97] under certified permission.

Now let us consider the Satisfaction of a Conjunctive Normal Form in propositional calculus. This problem is considered a NP problem (NP-complete as decision problem and NP-hard as solution when there are more than 2 literals for each clause).
In formal terms the problem is:

-Given a Conjunctive Normal Form, find an assignment for all literals (also called variables) that satisfies (i.e. renders true) the conjunction.
Obviously, the entire form is true if and only if all clause are true (i.e. satisfied). A clause is true if one of its literals is true. A literal can negated or not. The variable corresponding to a non-negated literal is true if the variable is assigned the value true and the value false for a negated literal.

An example of CNF (example 1) is:

(A + B).(C + D).(~B + ~C).(~A + ~D),

where + means OR, . AND, ~ NOT, (A + B), (C + D), (~B + ~C) and (~A + ~D) are the clauses, A and B are the literals for the first clause, etc.

A possible assignment that renders the form true is A = true, B = false, C = true, D = false.

We name v1, v2, etc. each clause (A + B), (C + D), etc.

-each clause must be satisfied: since a clause is a logical OR, it is sufficient for instance that A or B is true. Thus each clause vi must havean assignment among the available alternatives (i.e. the literals in that clause that are also called 'variables' because can receive a value of true or false)

-we use upper case letters for non-negated alternatives and lower case letters for negated alternatives.We justify this unconventional notation (b instead of ~B): we use the lower case for negated literals because we can use only one character for both negated and non negated literals. This fact simplified the computer programs that we used for solving the problems.

So we achieve:

v1: A, B
v2: C, D
v3: b, c
v4: a, d.

Of course, the choice of A (i.e. A = true) to satisfy the clause 1, does not permit the choice of NOT A that is the alternative a (i.e. A = false), for the clause 4. We cannot made incompatible choice.

For example, the following choice of literals to satisfy the clauses:

v1: A
v2: C
v3: b
v4: d

leads to:

A = true, C = true, B = false , D = false (because we have chosen the negated form of B and D).

There may be cases where the choices let undetermined some letter. In this case, both the assignments true and false are acceptable for that literal.

2.1. Conjunctive Normal Form Satisfaction, Integer and Linear Programming

We will show how to transform a m-CNF-SAT problem in an Integer Programming problem of the form

min cx
Ax = b, x > = 0, x integer,

with A integer matrix, b, c integer vectors. Moreover, all elements of A, b, c are 0 or 1. The solution of the integer LP problem is a valid solution of the CNF-SAT problem.

In [KAr72] taxonomy, the problem is represented as:

Our LP problem has a Matrix (and b e c vectors) with 0/1 values. This has strongly polynomial algorithms. See [Sch2021].
The solution of LP program is integer. [PaR88] reports the fundamental theorem:

If a rational constraint system (Ax >=b, x>=0) is total dual integer and if the vector b in integer, all extreme points solutions are integer. This is our case. For the complete proof see [ZiM97].

Note that this is one of the reasons why the present approach is so generally applicable: this constraint is not problem specific, it forces us to give an equal opportunity to each part of the problem data to be considered.
-(e) constraints which ensure that each pair literal such as A and a, B and b, etc. (i.e. non negated and negated forms) are mutual exclusive, that is at most one of two is 1. We call these constraints exclusion constraints.

Consider now an example with p=m=n=3. If we re-order the alternatives, we find:

A a B b C c

v1 y11 y12 y13 y14 y15 y16
v2 y21 y22 y23 y24 y25 y26
v3 y31 y32 y33 y34 y35 y36

Consider the following instance of the 3-variables case:

v1: A, B, C
v2: A, b
v3: a
the c row vector is: 1 0 1 0 1 0 1 0 0 1 0 0 0 1 0 0 0 0 000000000000000000

After a suitable Simplex procedure (i.e. with Pivot-operations for a suitable element in the first 3 rows), we obtain for the non-basis variables the zero-values, and for the basis variables:

x5=1, x10=1, x14=1 (non slack) => v1: C, v2: b, v3: a => A=B= false, C= true

v1: c; v2: a; v3: b.

Example 3:

4 original boolean variable, 3 clauses:

v1: A, B, C, D
v2: b, c, d
v3: a, B

Maximize p = x11 + x13 + x15 + x17 + x24 + x26 + x 28 + x32 subject to

x11 + x12 + x13 + x14 + x15 + x16 + x17 + x18 = 1

x21 + x22 + x23 + x24 + x25 + x26 + x27 + x28 = 1

x31 + x32 + x33 + x34 + x35 + x36 + x37 + x38 = 1

x11 + x22 <= 1

x12 + x21 <= 1
x13 + x24 <= 1
x14 + x23 <= 1
x15 + x26 <= 1
x16 + x25 <= 1
x17 + x28 <= 1
x18 + x27 <= 1

x11 + x32 <= 1
x12 + x31 <= 1
x13 + x34 <= 1
x14 + x33 <= 1
x15 + x36 <= 1
x16 + x35 <= 1
x17 + x38 <= 1
x18 + x37 <= 1

x21 + x32 <= 1
x22 + x31 <= 1
x23 + x34 <= 1
x24 + x33 <= 1
x25 + x36 <= 1
x26 + x35 <= 1
x27 + x38 <= 1
x28 + x37 <= 1

Example 5:

v1: A, B, C, D
v2: a,b,c
v3: d

Maximize p = x11 + x13 + x15 + x17 + x24 + x26 + x 28 + x38 subject to

x11 + x12 + x13 + x14 + x15 + x16 + x17 + x18 = 1

x21 + x22 + x23 + x24 + x25 + x26 + x27 + x28 = 1

x31 + x32 + x33 + x34 + x35 + x36 + x37 + x38 = 1

x11 + x22 <= 1

x12 + x21 <= 1
x13 + x24 <= 1
x14 + x23 <= 1
x15 + x26 <= 1
x16 + x25 <= 1
x17 + x28 <= 1
x18 + x27 <= 1

x11 + x32 <= 1
x12 + x31 <= 1
x13 + x34 <= 1
x14 + x33 <= 1
x15 + x36 <= 1
x16 + x35 <= 1
x17 + x38 <= 1
x18 + x37 <= 1

x21 + x32 <= 1
x22 + x31 <= 1
x23 + x34 <= 1
x24 + x33 <= 1
x25 + x36 <= 1
x26 + x35 <= 1
x27 + x38 <= 1
x28 + x37 <= 1

Solution: v1 (first clause) B; v2 (second clause) a; v3 (last clause) d

See the solution by ZweigMedia LLC free Simplex Solver.

Now consider the example 6

v1: A, B, C, D
v2: a,b,c
v3: D

Maximize p = x11 + x13 + x15 + x17 + x24 + x26 + x 28 + x37 subject to

x11 + x12 + x13 + x14 + x15 + x16 + x17 + x18 = 1

x21 + x22 + x23 + x24 + x25 + x26 + x27 + x28 = 1

x31 + x32 + x33 + x34 + x35 + x36 + x37 + x38 = 1

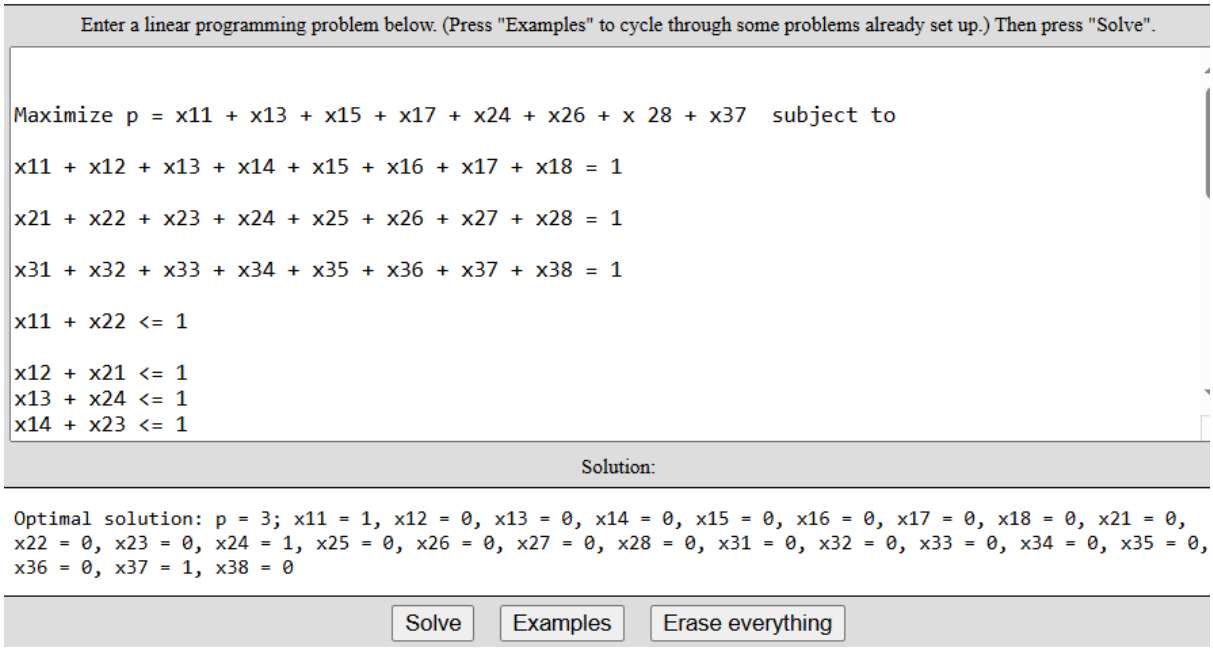
x11 + x22 <= 1

x12 + x21 <= 1
x13 + x24 <= 1
x14 + x23 <= 1
x15 + x26 <= 1
x16 + x25 <= 1
x17 + x28 <= 1
x18 + x27 <= 1

x11 + x32 <= 1
x12 + x31 <= 1
x13 + x34 <= 1
x14 + x33 <= 1
x15 + x36 <= 1
x16 + x35 <= 1
x17 + x38 <= 1
x18 + x37 <= 1

x21 + x32 <= 1
x22 + x31 <= 1
x23 + x34 <= 1
x24 + x33 <= 1
x25 + x36 <= 1
x26 + x35 <= 1
x27 + x38 <= 1
x28 + x37 <= 1

As one can see, the matrix A is the same, only the vector c is modified. This is the great advantage for our method.



Here is the first tableau

Tableau 1:

x11	**x12**	**x13**	**x14**	**x15**	**x16**	**x17**	**x18**	**x21**	**x22**	**x23**	**x24**	**x25**	**x26**
	x27	**x28**	**x31**	**x32**	**x33**	**x34**	**x35**	**x36**	**x37**	**x38**	**s1**	**s2**	**s3**

	s4	s5	s6	s7	s8	s9	s10	s11	s12	s13	s14	s15	s16
	s17	s18	s19	s20	s21	s22	s23	s24	s25	s26	s27	s28	s29
	s30	p											
*s1	1	1	1	1	1	1	1	1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	−1	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
*s2	0	0	0	0	0	0	0	0	1	1	1	1	1
	1	1	1	0	0	0	0	0	0	0	0	0	−1
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
*s3	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1	1	1	1	1	1	1	1	0	0
	−1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s4	1	0	0	0	0	0	0	0	0	1	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s5	0	1	0	0	0	0	0	0	1	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	1	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s6	0	0	1	0	0	0	0	0	0	0	0	1	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s7	0	0	0	1	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	1	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s8	0	0	0	0	1	0	0	0	0	0	0	0	0
	1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	1	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s9	0	0	0	0	0	1	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	1	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s10	0	0	0	0	0	0	1	0	0	0	0	0	0
	0	0	1	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s11	0	0	0	0	0	0	0	1	0	0	0	0	0
	0	1	0	0	0	0	0	0	0	0	0	0	0

	0	0	0	0	0	0	0	0	1	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s12	1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	1	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	1	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s13	0	1	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s14	0	0	1	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	1	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	1	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s15	0	0	0	1	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	1	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s16	0	0	0	0	1	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	1	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s17	0	0	0	0	0	1	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s18	0	0	0	0	0	0	1	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	1	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s19	0	0	0	0	0	0	0	1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	1	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s20	0	0	0	0	0	0	0	0	1	0	0	0	0
	0	0	0	0	1	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	1	0	0	0	0	0	0	0	0
	0	0	0	1									
s21	0	0	0	0	0	0	0	0	0	1	0	0	0
	0	0	0	1	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	1	0	0	0	0	0	0	0
	0	0	0	1									
s22	0	0	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	1	0	0	0	0	0	0

	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	1	0	0	0	0	0	0
	0	0	0	1									
s23	0	0	0	0	0	0	0	0	0	0	0	1	0
	0	0	0	0	0	1	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	1	0	0	0	0	0
	0	0	0	1									
s24	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	0	1	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	1	0	0	0	0
	0	0	0	1									
s25	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	0	0	0	1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	1	0	0	0
	0	0	0	1									
s26	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	1									
s27	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	1	0	0	0	0	0	0	1	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	1	0
	0	0	0	1									
s28	1	1	1	1	1	1	1	1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	1									
s29	0	0	0	0	0	0	0	0	1	1	1	1	1
	1	1	1	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	1									
s30	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1	1	1	1	1	1	1	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	0	1									
p	−1	0	−1	0	−1	0	−1	0	0	0	0	−1	0
	−1	0	−1	0	0	0	0	0	0	−1	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	1	0									

Tableau 4:

x11	x12	x13	x14	x15	x16	x17	x18	x21	x22	x23	x24	x25	x26
	x27	x28	x31	x32	x33	x34	x35	x36	x37	x38	s1	s2	s3
	s4	s5	s6	s7	s8	s9	s10	s11	s12	s13	s14	s15	s16

	s17	s18	s19	s20	s21	s22	s23	s24	s25	s26	s27	s28	s29
	s30	p											
s1	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	1	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	0									
s2	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0									
x31	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1	1	1	1	1	1	1	1	0	0
	−1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s4	0	−1	−1	−1	−1	−1	−1	−1	0	1	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	−1
	0	0	0	0									
s5	0	1	0	0	0	0	0	0	0	−1	−1	−1	−1
	−1	−1	−1	0	0	0	0	0	0	0	0	0	0
	0	0	1	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	−1	0	0	0									
s6	0	0	1	0	0	0	0	0	0	0	0	1	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s7	0	0	0	1	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	1	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s8	0	0	0	0	1	0	0	0	0	0	0	0	0
	1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	1	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s9	0	0	0	0	0	1	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	1	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s10	0	0	0	0	0	0	1	0	0	0	0	0	0
	0	0	1	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s11	0	0	0	0	0	0	0	1	0	0	0	0	0
	0	1	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	1	0	0	0	0

	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s12	0	−1	−1	−1	−1	−1	−1	−1	0	0	0	0	0
	0	0	0	0	1	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	1	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	−1
	0	0	0	0									
s13	0	1	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	−1	−1	−1	−1	−1	−1	−1	0	0
	1	0	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0									
s14	0	0	1	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	1	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	1	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s15	0	0	0	1	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	1	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s16	0	0	0	0	1	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	1	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s17	0	0	0	0	0	1	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s18	0	0	0	0	0	0	1	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	1	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s19	0	0	0	0	0	0	0	1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	1	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s20	0	0	0	0	0	0	0	0	0	−1	−1	−1	−1
	−1	−1	−1	0	1	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	1	0	0	0	0	0	0	0	0
	−1	0	0	0									
s21	0	0	0	0	0	0	0	0	0	1	0	0	0
	0	0	0	0	−1	−1	−1	−1	−1	−1	−1	0	0
	1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	1	0	0	0	0	0	0	0
	0	0	0	0									
s22	0	0	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	1	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0

	0	0	0	0	0	0	1	0	0	0	0	0	0
	0	0	0	1									
s23	0	0	0	0	0	0	0	0	0	0	0	1	0
	0	0	0	0	0	1	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	1	0	0	0	0	0
	0	0	0	1									
s24	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	0	1	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	1	0	0	0	0
	0	0	0	1									
s25	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	0	0	0	1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	1	0	0	0
	0	0	0	1									
s26	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	1									
s27	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	1	0	0	0	0	0	0	1	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	1	0
	0	0	0	1									
x11	1	1	1	1	1	1	1	1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	1									
x21	0	0	0	0	0	0	0	0	1	1	1	1	1
	1	1	1	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	1									
s30	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	0	0									
p	0	1	0	1	0	1	0	1	0	0	0	−1	0
	−1	0	−1	0	0	0	0	0	0	−1	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	1	1									

Tableau 5:

x11	x12	x13	x14	x15	x16	x17	x18	x21	x22	x23	x24	x25	x26
	x27	x28	x31	x32	x33	x34	x35	x36	x37	x38	s1	s2	s3
	s4	s5	s6	s7	s8	s9	s10	s11	s12	s13	s14	s15	s16
	s17	s18	s19	s20	s21	s22	s23	s24	s25	s26	s27	s28	s29
	s30	p											

s1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**
	0	**0**	**0**	**0**									
s2	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	1	**0**	**0**	**0**									
x31	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**
	−1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**1**									
s4	**0**	**−1**	**−1**	**−1**	**−1**	**−1**	**−1**	**−1**	**0**	**1**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**−1**
	0	**0**	**0**	**0**									
s5	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**1**									
s6	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**−1**	**−1**	**−1**	**0**	**−1**
	−1	**−1**	**−1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	−1	**0**	**0**	**0**									
s7	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	
	0	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	
	0	**0**	**0**	**1**									
s8	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	
	1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	
	0	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	
	0	**0**	**0**	**1**									
s9	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**1**									
s10	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**1**									
s11	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**
	0	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**1**									

s12	**0**	**−1**	**−1**	**−1**	**−1**	**−1**	**−1**	**−1**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**−1**
	0	**0**	**0**	**0**									
s13	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**−1**	**−1**	**−1**	**−1**	**−1**	**−1**	**−1**	**0**	**0**
	1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**									
s14	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**1**									
s15	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**1**									
s16	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**1**									
s17	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**1**									
s18	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**1**									
s19	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**1**									
s20	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**1**									
s21	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**−1**	**−1**	**−1**	**−1**	**−1**	**−1**	**−1**	**0**	**0**
	1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**									
s22	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**1**									

s23	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**−1**	**−1**	**−1**	**0**	**−1**
	−1	**−1**	**−1**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**
	−1	**0**	**0**	**0**									
s24	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**1**									
s25	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	1	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
	0	**0**	**0**	**1**									
s26	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
	0	**0**	**0**	**1**									
s27	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**
	0	**0**	**0**	**1**									
x11	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**
	0	**0**	**0**	**1**									
x24	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**
	1	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	1	**0**	**0**	**1**									
s30	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**1**	**0**	**0**									
p	**0**	**1**	**0**	**1**	**0**	**1**	**0**	**1**	**1**	**1**	**1**	**0**	**1**
	0	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**−1**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**
	1	**0**	**1**	**2**									

Tableau 8:

x11	**x12**	**x13**	**x14**	**x15**	**x16**	**x17**	**x18**	**x21**	**x22**	**x23**	**x24**	**x25**	**x26**
	x27	**x28**	**x31**	**x32**	**x33**	**x34**	**x35**	**x36**	**x37**	**x38**	**s1**	**s2**	**s3**
	s4	**s5**	**s6**	**s7**	**s8**	**s9**	**s10**	**s11**	**s12**	**s13**	**s14**	**s15**	**s16**
	s17	**s18**	**s19**	**s20**	**s21**	**s22**	**s23**	**s24**	**s25**	**s26**	**s27**	**s28**	**s29**
	s30	**p**											
s1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**

	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	0									
s2	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0									
s3	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	0	0									
s4	0	−1	−1	−1	−1	−1	−1	−1	0	1	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	−1
	0	0	0	0									
x26	0	0	0	0	0	0	0	0	1	1	1	0	1
	1	1	1	0	0	−1	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	−1	0	0	0	0	0
	1	0	0	0									
s6	0	0	1	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	−1	0	0	0	0	0	0	0
	0	0	0	1	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	−1	0	0	0	0	0
	0	0	0	0									
s7	0	0	0	1	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	1	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s8	0	0	0	0	1	0	0	0	−1	−1	−1	0	−1
	0	−1	−1	0	0	1	0	0	0	0	0	0	0
	0	0	0	0	0	1	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	1	0	0	0	0	0
	−1	0	0	1									
s9	0	0	0	0	0	1	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	1	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s10	0	0	0	0	0	0	1	0	0	0	0	0	0
	0	0	1	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s11	0	0	0	0	0	0	0	1	0	0	0	0	0
	0	1	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	1	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s12	0	−1	−1	−1	−1	−1	−1	−1	0	0	0	0	0
	0	0	0	0	1	0	0	0	0	0	0	0	0

	0	0	0	0	0	0	0	0	0	1	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	−1
	0	0	0	0									
s13	0	1	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s14	0	0	1	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	1	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	1	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s15	0	0	0	1	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	1	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s16	0	0	0	0	1	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	1	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s17	0	0	0	0	0	1	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s18	0	0	0	0	0	0	1	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	1	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s19	0	0	0	0	0	0	0	1	0	0	0	0	0
	0	0	0	−1	−1	−1	−1	−1	−1	0	−1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1	0	0	0	0	0	0	0	0	0
	0	−1	0	0									
s20	0	0	0	0	0	0	0	0	1	0	0	0	0
	0	0	0	0	1	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	1	0	0	0	0	0	0	0	0
	0	0	0	1									
s21	0	0	0	0	0	0	0	0	0	1	0	0	0
	0	0	0	1	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	1	0	0	0	0	0	0	0
	0	0	0	1									
s22	0	0	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	1	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	1	0	0	0	0	0	0
	0	0	0	1									
s5	0	1	0	0	0	0	0	0	1	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0

	0	0	1	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1									
s24	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	0	1	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	1	0	0	0	0
	0	0	0	1									
s25	0	0	0	0	0	0	0	0	−1	−1	−1	0	−1
	0	−1	−1	0	0	1	0	1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	1	0	1	0	0	0
	−1	0	0	1									
s26	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	1	0	0
	0	0	0	1									
s27	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	1	−1	−1	−1	−1	−1	−1	0	−1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	1	0
	0	−1	0	0									
x11	1	1	1	1	1	1	1	1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	0	0	1									
x24	0	0	0	0	0	0	0	0	0	0	0	1	0
	0	0	0	0	0	1	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	1	0	0	0	0	0
	0	0	0	1									
x37	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	1	1	1	1	1	1	1	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	0	1									
p	0	1	0	1	0	1	0	1	1	1	1	0	1
	0	1	0	1	1	1	1	1	1	0	1	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	1
	1	1	1	3									

p = 3; x11 = 1, x12 = 0, x13 = 0, x14 = 0, x15 = 0, x16 = 0, x17 = 0, x18 = 0, x21 = 0, x22 = 0, x23 = 0, x24 = 1, x25 = 0, x26 = 0, x27 = 0, x28 = 0, x31 = 0, x32 = 0, x33 = 0, x34 = 0, x35 = 0, x36 = 0, x37 = 1, x38 = 0

Note that it was not requred an integer solution by means of an integer solver but the standard simplex always finds integer solutions. See [ZiM97] for the proof.

Now consider the example 7

v1: D
v2: a,b,c
v3: d

Example 7

Maximize p = x18 + x24 + x26 + x 28 + x37 subject to

x11 + x12 + x13 + x14 + x15 + x16 + x17 + x18 = 1

x21 + x22 + x23 + x24 + x25 + x26 + x27 + x28 = 1

x31 + x32 + x33 + x34 + x35 + x36 + x37 + x38 = 1

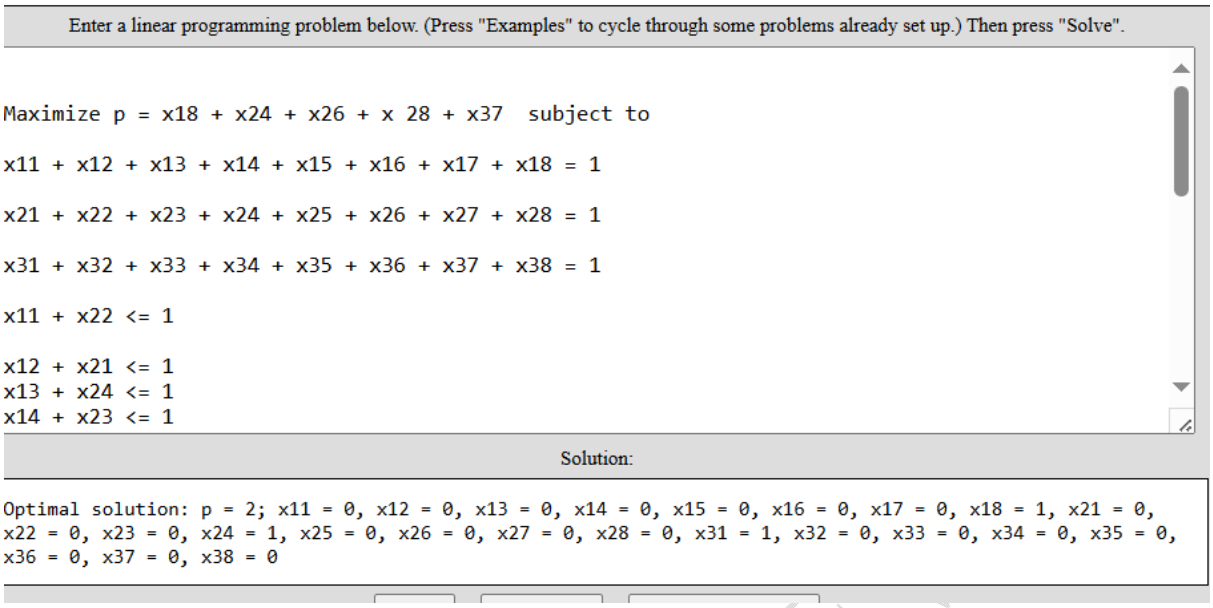
x11 + x22 <= 1

x12 + x21 <= 1
x13 + x24 <= 1
x14 + x23 <= 1
x15 + x26 <= 1
x16 + x25 <= 1
x17 + x28 <= 1
x18 + x27 <= 1

x11 + x32 <= 1
x12 + x31 <= 1
x13 + x34 <= 1
x14 + x33 <= 1
x15 + x36 <= 1
x16 + x35 <= 1
x17 + x38 <= 1
x18 + x37 <= 1

x21 + x32 <= 1
x22 + x31 <= 1
x23 + x34 <= 1
x24 + x33 <= 1
x25 + x36 <= 1
x26 + x35 <= 1
x27 + x38 <= 1
x28 + x37 <= 1

that is infeasible.



Our solver finds a partial solution (maximum SAT).

Brief summary of tests

Variables	Constraints	Iterations for the Simplex	
12	6	5	CNFSAT2 2 booleans, $2^{\wedge 2}$ = 4 alternatives
18	21	8	CNFSAT3 (NP) 3 boolleans, 8
24	27	10	CNFSAT4 (NP) 4 booleans 16
120	783	40	CNFSAT20 (NP) 20 booleans $2^{\wedge 20}$ = 1048576
180	16113	60	CNFSAT30 (NP) 30 booleans 1073741824

etc.

In general, for our experiments (1000 done), the number of iterations grows linearly with the number of input data.

The Rutgers University' Center for Discrete Mathematics (DIMACS) maintained a data base of very hard SAT problems and a problem generator, we used for our tests. For a detailed report on experimental data, see [ZiM97].

2.3. Algorithm Cost

The worst case cost in the dimensions [p * n] of the original CNF-SAT problem is

number of columns: p * 2 * n + 2 * n * p * (p - 1) / 2
number of rows : p + 2 * n * p * (p - 1) / 2.

If m=n=p, we have:
c = n 3 + n 2
r = n 3 - n 2 + n

giving a cubic Worst Case Cost.

2.4. Acceerating the solution by means of intelligent pivot election.

We have tested Artificial Neural Networks that learn to choose the position of pivot operation in the Simplex in order to reduce the number of iterations. The first approach is described in [Mon98]. We considered several neural networks paradigms. The best result was achieved with Functional-Link Fast Backpropagation Network (FL-F-BKP) with 2 * n^2 elements in the hidden layer.

The functional-link network FLN) is a feedforward network that uses backpropagation algorithm.We used the outer product tensor model variant.

Architecture for FLN:

Input layer. 2 * n^2 PEs (processing elements)

Output layer: 2 * n^2 Pes.

We implemted our networs in Python augmented through NumPy, SciPy, scikit-lear, Mathplothlib and pandas libraries [Mon2015].

Brief summary of tests: average speed up:

Variables	Constraints	Iterations for the Simplex	
24	27	8	CNFSAT4 (NP) 4 booleans 16 alternatives
120	783	30	CNFSAT20 (NP) 20 booleans $2^{\wedge 20}$ = 1048576
180	16113	50	CNFSAT30 (NP) 30 booleans 1073741824

etc.

2.5. Other Neural Network approaches.

[ChL2022] describes a novel neural network approach for constraint optimization that uses a Neural Optimization Machine (NOM) .

[KhA2023] presents a variant of a recurrent neural network (RNN) with variational classical annealing (VCA)-

3. The Traveling Salesperson Problem (TSP)

We report here from [ZiM97] the same approach of the previous sections to solve the famous Traveling Salesperson Problem (TSP), by means of Linear Programming.

We have been choosing a problem apparently very different from m-CNF-SAT to show that a very uniform modelling technique can be successfully used.

The well known TSP is that of a salesperson which has a list of pcities, each of which he must visit exactly once. There are direct roads between each pair of cities on the list. We must find the route the salesperson should follow so that he travels the shortes possible distance on a round trip, starting at any one of the cities and then returning there: the problem is in fact NP-hard.

In graph theory, a Hamiltonian circuit in a graph is a closed walk that visit each vertex exactly once and is the model for TSP. Deciding whether or not a graph has a Hamitonian circuit is an NP complete problem, capturing much of the complexity of the general TSP.

There is an important connection between traveling and map coloring: the boundaries of a map's regions are the edges of a graph, with the intersection points as vertices.

The dual of Hamiltonian circuit is Euler circuit: travel each edge exactly once. However this problem is well solved, it is of P class of complexity. Euler circuit can be the model of Shared Resource Allocation, see [ZiM97].
CNFSAT problem can be modelled as an Euler circuit but in hypergraphs, and this has NP complexity. A hypergraph has and edge connecting more than two vertices unlike a graph.

The TSP has several applications such as planning, logistics, microchips factory, DNA sequencing, astronomy, optimal control.

The constraints:

c') multiple choice constraints which ensure that exactly one of several costs is chosen at any step; there are p constraints for p cities.

c'') constraints that ensure at most one 1-value for each column is present, i.e. a cost is chosen at most one time for the entire route. There are p2 - p constraints.

e) constraints that ensure the salesperson not to return to the start-city before the entire route is completed and ensure continuity (if we choose c12, we can not then choose c3...),

See [ZiM97].

4. Proof for our CNF-SAT solver

Complete proof of Corretcness and Completeness for our LP solver can be found in [ZiM97].

Conclusions.

The recent result of [Sch2021] renders very important our approach and solver for CNFSAT and other NP hard problems [ZiM97]. Our LP Simplex solver for CNF-SAT (NP-complete) and other NP hard problems, has a Matrix (and b e c vectors) with 0/1 values. This has strongly polynomial algorithms. See [Sch2021]. We tested our LP Sover with hard cases of problems, and reported the experimental results. Thus the present paper demonstrates in practice the claim of [Sch2021].

For other Constraint Satisfaction Problems see [Mon92,97,98,2011, 2013, 2015,2023], [ZiM97].

References.

[ChL2022]
-Chen J., Y. Liu, Neural Optimization Machine, arXiv 2208.03891, 2022
[Coo71]
-Cook S.A., The Complexity of Theorem Proving Procedures, Proc. 3rd ann.ACM Symp.Theory Comput.(1971)
[DaP60]
-Davis M., H. Putnam, A computing procedure for quantification theory, J. ACM 8, 1960, pp. 201-215
[FrP83]
-Franco J., M. Paull, Probabilistic analysis of the Davis-Putnam procedure for solving the satisfiability problem, Discr. Appl. Math. 5, 1983, pp. 77-87
[GaJ79]
-Garey M.R. e D.S. Johnson, Computer and Intractability, (Freeman, San Francisco, 1979)
[GeW94]
-Gent I.P, and T. Walsh, Easy problems are sometimes hard, AI Vol. 70, N.1-2, Elsevier, 1994
[HoK56]
-Hoffman A.J., J.B. Kruskam, Integral Boundary Points of Convex Polyhedra, in Linear Inequalities an Related Systems (H.W. Kuhn and A.W. Tucker eds), Princeton University Press, Princeton, 1956.
[HoK73]
-Hopcroft, R. Karp, The matching problem for bipartite graphs, in SIAM J. Comput.,(1973)
[JeW90]
-Jeroslow R.G., J. Wang, Solving propositional satisfiability problems, in Annals of Mathematics and Artificial Intelligence, M.C. Golumbic editor, J.C. Baltzer Scientific Publishing Company, Basel, Switzerland, 1990
[Kar84]
-Karmarkar N, A New Polynomial Time Algorithm for Linear Programming, in Proceedings of the 16th Annual ACM Symposium on Theory of Computing, 1984
[KAr72]
-Karp R.M., Reducibility Among Combinatorial Problems, in Complexity of Computer Computations (R.E. Miller and J.W. Thatcher, eds), Plenum Press, New York, 1972
[Kha79]
-Khachian L.G., A Polynomial Algorithm for Linear Programming, Soviet Mathematics Dklady, Nauk USSR, 1979
[KhA2023]
-Khandoker S.A., J. M. Abedin, M. H. Hibat-Allah, Supplementing recurrent networks with annealing to combinatorial optimization problems, Sci Technical 4, 2023

[Lap93]
-Laporte G., Recent algorithmic developments for the Traveling Salesman problem and the Vehicle Routing problem, Ricerca Operativa, Franco Angeli pub., Milano, N. 68, 1993, pp. 5-28
[Lov78]
-Loveland D.W., Automated Theorem Proving: a Logical Basis, North-Holland, Amsterdam, 1978
[Mac92]
-Mackworth A.K., The logic of constraint satisfaction, A.I., Vol. 58, N. 1-3, (1992) 3-20
[MaF92]
-Mackworth A.K., E.C. Freuder, editors of the special volume on Constraint-Based Reasoning, A.I., Vol. 58, N. 1-3 (1992)
[MSL92]
-Mitchell D., B. Selman, and H. Levesque, Hard and easy distributions for SAT problems, in Proceedings of the Tenth National Conference on Artificial Intelligence, (AAAI-92), 1992, pp. 459-465
[Mon86]
-Monfroglio A., School time table scheduling in prolog, ACM SIGART Bulletin, 20-22 logic constraint satisfaction, Atti Congresso Annuale AICA, 167-182 (proc. AICA Conference, 1989
[Mon89]
-Monfroglio A., Efficient
[Mon92]
-Monfroglio A., Integer programs for logic constraint satisfaction, Theoretical Computer Science, the Journal of the EATCS, North-Holland, vol. 100, 1992
[Mon92b]
-Monfroglio A., Hybrid Genetic Algorithms for Constraint Satisfaction, Proceedings A.I.C.A. Conference, Turin, Italy (1992) 803--
[Mon93]
-Monfroglio A., Logic decisions under constraints, Decision Support Systems, North-Holland, V.11, 1993
[Mon98]
-Monfroglio A., Finite Constraint Satisfaction, C.T. Leondes, Algorithms and Architectures, Academic Press, 1998
[Mon2011]
-Monfroglio A., Didamatica AICA, Politecnico di Torino, 2011
[Mon2013]
-Monfroglio A., Didamatica AICA, CNR Pisa, 2013
[Mon2015]
-Monfroglio A., Didamatica AICA, Università di Genova, 2015
[Mon2023]
-Monfroglio A., Polynomial Time Algorithms for Hard Constraint Satisfaction Problems, SSRN 4436859, (2023)
[Nev74]
-Nevins A.J., A human oriented logic for automatic theorem-proving, J.ACM 21, 1974, pp. 606-621
[PaY82]
-Papadimitriou C.H., M. Yannakakis, The Complexity of Facets (and Some Facets of Complexity), in Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing, SanFrancisco, 1982
[PaR88]
-Parker R.G., R.L.Rardin, Discrete Optimization,Academic Press, San Diego, CA, 1988
[Rei94]
-Reinelt G., The traveling Salesman, Springer-Verlag, Berlin, 1994
[SaM89]
-Salkin H. M., K. Mathur, Foundations of Integer Programming, North-Holland, New York, 1989
[Sch2021]
-Schrijver A., Theory of Linear & Integer Programming, Wiley & Sons, 2021
[ScS19]
-Schiex T., S. de Givry Eds, Principles and Practice of Constraint Programming, Springer, Proc. CP 2019

[Sey80]
-Seymour P.D., Decomposition of Regular Matroids, Journal of Combinatorial Theory, 1980
[Tar86]
-Tardos E., A strongly polynomial algorithms to solve combinatorial linear programs. Operations Research 34 (1986)
[Van89]
-Van Hentenryck P., Constraint Satisfaction in Logic Programming, MIT PRESS, 1989
[VeD68]
-Veinott A.F., G.B. Dantzig, Integral Extreme Points, SIAM Review, 10, 1968. .
[YiY90]
-Ye Yinyu, A 'Build-down scheme for Linear Programming, Mathematical Programming, Vol. 46, North-Holland, Amsterdam, 1990, pp. 61-72
[ZiM97]
-Zimmermann H.J., A. Monfroglio, Linear programs for constraint Satisfaction problems, EJOR, Elsevier, 97 (1997).