EasyChair Preprint

№ 3444

An Integrated Linear Programming Approach in Optimization of Crops

Arindam Biswas, Priti Thadani, Ankita Dikshit, Sukla Atha, Ananya Biswas, Ansh Gupta, Chayan Halder and Anirban Das

May 19, 2020

AN INTEGRATED LINEAR PROGRAMMING APPROACH IN OPTIMIZATION OF CROPS

Arindam Biswas
Dept. of CSE
UEM Kolkata, India
Email:
arindambiswas186@gmail.com

Priti Thadani
Dept. of CSE
UEM Kolkata, India
Email:
pritithadani43@gmail.com

Ankita Dikshit
Dept. of CSE
UEM Kolkata, India
Email:
ankitadikshit946@gmail.com

Sukla Atha
Dept. of CSE
UEM Kolkata, India
Email: sukla.atha52@gmail.com

Ananya Biswas
Dept. of CSE
UEM Kolkata, India
Email:
ananya2201@gmail.com

Ansh Gupta
Dept. of CSE
UEM Kolkata, India
Email:
ansh.gupta.pakur@gmail.com

Chayan Halder
Dept. of CSE
UEM Kolkata, India
Email:
chayan.halder@uem.edu.in

Anirban Das
Dept. of CSE
UEM Kolkata, India
Email:
anirban-das@live.com

ABSTRACT **- Crop planning is a Multi-objective optimization problem, also called as NP-Hard Problem. Agriculture has to face various challenges such as irrigation water management, land allocation, climate changes, human and other resources used for agriculture, which can be controlled by effective crop planning. Crop planning optimization gives an idea to utilize the minimum resources to get the maximum profit through optimizing the objectives. Agriculture plays an important role in the ecosystem management; using bio-fertilizer and green manure can improve soil fertility without any chemical effects. The main aim of the crop planning is to improve the profit and productivity with the low input cost and resources. The crop planning problem has many factors, in that some can be optimized and some cannot be optimized. There are many methods available to solve crop planning problem, like algorithms, optimization tools, decision making tools, software etc. but still it needs more improvement to get the best optimal solutions. This paper presents a general idea of Crop Planning and various algorithms which are used to solve this problem and also address the problems that need effective solutions.**

Keywords **– Optimization, Boolean, minisat, linear programming.**

I. INTRODUCTION

The engineers have to carry out standard calculations for evapotranspiration, crop water usage, fertilizers, manure cost etc. Hence recommendations can be made with respect to improved practices in irrigation and better planning of irrigation schedules. Smart application of fertilizer illustrates payoff in using analytical tools to enhance crop yields and improve the environment.

The fertilizer used for agriculture is 40% globally. This has resulted in new methods of saving money in buying fertilizers. Hence money saving techniques has to be practiced. In fertilizer resources planning and management, optimization techniques is used for limited use of resources such as such as water, land, production cost, manpower, fertilizers, seeds, and pesticides. For cultivating each crop, the land area needs to be planned properly. Hence the crop pattern has to be decided optimally depending on available water and mineral resources and on economic basis. Therefore farmer needs to be educated to adopt optimum cropping pattern

which maximizes the economic returns. Hence the study is taken up to optimize the fertilizers used for crops. The objective function for multi crop model were formulated using linear programming for
Maximizing the net benefits.

To plan the with regard to distribution of water resources to the crop, it is important to optimize the available land and water Resources to achieve maximum returns. To solve such problems, the mathematical programming models like linear programming (LP), dynamic programming (DP) and genetic algorithm (GP) are used. Linear programming is the most convenient and effective tool to handle more number of constraints.

in agricultural research and farming systems . Linear 10
Linear programming is one of the main Operations Research techniques. The mathematical model usually consists of linear equations and/or inequalities. There is a linear objective function that is optimized (maximized or minimized) which is subject to a set of constraints. Generically, a linear programming model applied to farm planning .

II. STUDY AREA

The objective function of the model is subject to the following constraints: water availability, crop land requirement, Human labour cost, Animal and Machine power cost, Seed cost, Fertilizers and Manure cost, Fixed cost etc.

An **integer programming** problem is a mathematical optimization or feasibility program in which some or all of the variables are restricted to be integers. In many settings the term refers to **integer linear programming** (ILP), in which the objective function and the constraints (other than the integer constraints) are linear.

Integer programming is NP-complete. In particular, the special case of 0-1 integer linear programming, in which unknowns are binary, and only the restrictions must be satisfied, is one of Karp's 21 NP-complete problems.

If some decision variables are not discrete the problem is known as a **mixed-integer programming** problem

III. APPLICATIONS

There are two main reasons for using integer variables when modelling problems as a linear program:

1. The integer variables represent quantities that can only be integer. For example, it is not possible to build 3.7 cars.
2. The integer variables represent decisions (e.g. whether to include an edge in a graph) and so should only take on the value 0 or 1.

These considerations occur frequently in practice and so integer linear programming can be used in many applications areas, some of which are briefly described below.

a) Production planning

Mixed integer programming has many applications in industrial production, including job-shop modelling. One important example happens in agricultural production planning involves determining production yield for several crops that can share resources (e.g. Land, labour, capital, seeds, fertilizer, etc.). A possible objective is to maximize the total production, without exceeding the available resources. In some cases, this can be expressed in terms of a linear program, but variables must be constrained to be integer.

b) Scheduling

These problems involve service and vehicle scheduling in transportation networks. For example, a problem may involve assigning buses or subways to individual routes so that a timetable can be met, and also to equip them with drivers. Here binary decision variables indicate whether a bus or subway is assigned to a route and whether a driver is assigned to a particular train or subway. The zero-one programming technique has been successfully applied to solve a project selection problem in which projects are mutually exclusive and/or technologically interdependent. It is used in a special case of integer programming, in which all the decision variables are integers. It can assume the values either as zero or one.

IV. EXACT ALGORITHMS

When the matrix is not totally unimodular, there are a variety of algorithms that can be used to solve integer linear programs exactly. One class of algorithms are cutting plane methods which work by solving the LP relaxation and then adding linear constraints that drive the solution towards being integer without excluding any integer feasible points.

Another class of algorithms are variants of the branch and bound method. For example, the branch and cut method that combines both branch and bound and cutting plane methods. Branch and bound algorithms have a number of advantages over algorithms that only use cutting planes. One advantage is that the algorithms can be terminated early and as long as at least one integral solution has been found, a feasible, although not necessarily optimal, solution can be returned. Further, the solutions of the LP relaxations can be used to provide a worst-case estimate of how far from optimality the returned solution is. Finally, branch and bound methods can be used to return multiple optimal solutions.

The pseudo-Boolean (PB) solver npSolver encodes PB into SAT and solves the optimization instances by calling a SAT solver iteratively. The system supports MaxSAT, PB and WBO. Optimization instances are tackled by a greedy lower bound mechanism first. The solver can translate PB to SAT based on a portfolio of different encodings, based on the number of clauses. As back end of the system any SAT solver can be used, even incremental solvers for the optimization function. By using glucose as back end and the SAT simplifier Coprocessor, npSolver shows an outstanding performance on decision instances and can compete on optimization instances.

Even though the SAT problems are said to be NP-complete, there has been a enormous enhancement in the technology of SAT solver over the years. The improvement leads to the evolution of various efficient SAT algorithms which has the capability of solving different problems having millions of constraints and thousands of variables. Several well-known solvers are like zChaff , RSat , GRSAP , and Berkmin etc. Due to an enormous improvement in the field of Boolean satisfiability (SAT), the SAT solver increased it's capability to handle pseudo-Boolean (PB) constraints. It can solve the 0-1 Integer linear programming problems with great efficiency. A large number of complex techniques from different engineering and science domain have used the advanced Boolean SAT solver to solve various problems. The problems are like scheduling, routing , power minimization, cryptography, MANETs and many more. Several researchers have been used to solve the crop optimization problems using Boolean Satisfiability.

In this paper, a complete solution is presented to solve the crop optimization problem using pseudo-Boolean satisfiability algorithm. The difference between Boolean and pseudoBoolean is that pseudo-Boolean constraints can be represented as linear inequalities with integer coefficients. PB constraints performs efficiently in representing "counting constraints". Moreover, the PB-SAT solvers PBS , Pueblo, Bsolo and MiniSAT+ can handle both decision problems and optimization problems efficiently.

V. PROBLEM FORMULATION

The proposed methodology illustrates a PB-SAT based formulation for crop optimization problem. Given a list of different brand of fertilizers with different level of nitrogen and phosphate with different cost requirement, the objective is to find the minimum cost for fertilizer required with satisfactory amount of nitrogen and phosphate for different crops such that the total purchase cost of fertilizers gets reduced satisfying all the given constraints. Some of the assumptions and notations used to design the pseudo-Boolean constraint formulation is listed below:

Table 1:Assumptions

Sr.no.	Brand	Cost (per bag)	Nitrogen (lb/bag)	Phosphate (lb/bag)
1.	Super-go	100	2	4
2.	Crop-quick	120	4	3

LP Model Formulation

The following assumptions are considered while designing the constraints of the farming.

- Two brands of fertilizers are available Super-go, Crop-quick.
- Field requires at least 16 pounds of nitrogen and 24 pounds of phosphate.
- Super-gro costs $6 per bag, Crop-quick $3 per bag.
- Problem: How much of each brand to purchase to minimize total cost of fertilizer given following data?

Table 2. Chemical Contribution

Brand	**Chemical Contribution**	
	Nitrogen (lb/bag)	Phosphate (lb/bag)
Super-go	2	4
Crop-quick	4	3

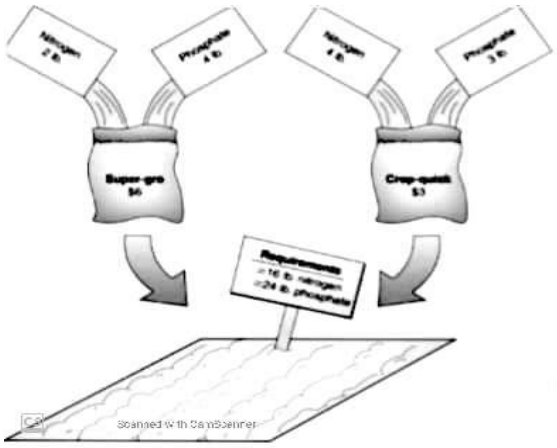

Figure 1. Farmer's Field 1

PB-formulation: The PB-formulations designed for the personnel scheduling problem is discussed below:

a) **Fertilizer selection constraint**:

The constraint illustrates that Fertilizer selection of a particular day must be constrained by the nutrient requirement of the corresponding day. As the scheduling is designed for any two consecutive fertilizer of a month So, the Super-Grow covers two consecutive intervals. But, Crop-Quick using tenure considers only one interval at a time. The above considerations are formulated through the following PB-constraint:

The notations required to design the PB-constraints are given below:

Table 3.Notations

Notation	**Description**
Di,j	denotes an integer variable representing the fertilizer requirement on day i interval j.
N	denotes highest number of digits used to represent the integer variable Di,j in Boolean form.
Xi,j,b	represents a Boolean variable used to denote a "Super-Gro" used in day i interval j
Xpi,pj,b	represents a Boolean variable used to denote a "Super-Gro" continuing use from previous interval pj of day pi.
Yi,j,b	presents a Boolean variable used to define a "Crop-Quick" used in day i interval j
CostSG	total cost incurred by a "Super-Grow"
CostCQ	total cost incurred by a "Crop-quick"

$$\sum_{b=0}^{n} 2^b * X_{i,j,b} + \sum_{b=0}^{n} 2^b * X_{pi,pj,pb} + \sum_{b=0}^{n} 2^b * Y_{i,j,b} = D_{i,j} \quad (1)$$

b) **Fertilizer amount constraint**: This constraint ensures that at least one Nutrient

must be present from each category for any crop. The constraint is applied for both Nitrogen and Phosphate as we have considered that every category crop have some specific specialization. The smallest number of nutrient assignment of any type could be one. The constraint can be represented as defined below: The PB-formulation for Nitrogen content:

$$\sum_{b=0}^{n} 2^b * X_{i,j,b} + \sum_{b=0}^{n} 2^b * X_{pi,pj,b} \geq 1 \quad (2)$$

The PB-formulation for Phosphate content:

$$\sum_{b=0}^{n} 2^b * Y_{i,j,b} \geq 1 \quad (3)$$

Objective function The above two constraints ensures the fertilizer constraints, which ensure that the nutrients get allotted to each crop with zero inconsistency. Although, it is unable to minimize the fertilizer cost during allocation. To implement this, the following objective function is designed:

$$\text{Min: } \text{Cost}_{ni} * \left(\sum_{b=0}^{n} 2^b * X_{i,j,b} \right) + \text{Cost}_{ph} * \left(\sum_{b=0}^{n} 2^b * Y_{i,j,b} \right) \quad (4)$$

EXAMPLE DISCUSSION

In this section, an example is presented where two types of fertilizers are available. The fertilizers are categorized based on their nitrogen and phosphate content. The objective is to obtain the fertilizer with least cost and higher content of nitrogen and phosphate such that we get higher output of crops at least expenditure. The nutrient required for any particular fertilizer is presented in Table I. For example, the super-go with nitrogen … and phosphate … is represented by X1. The Boolean representation of X1 and X2 is presented in Fig. 1. In this example, the highest number of personnel requirement is 11 and 4 digits are required to represent the integer value into Boolean form. So, X1 is represented by 4 different Boolean variables. The total optimization costs are considered different based on the fertilizers. The Super-gro have paid at $6 per bag, and Crop-quick are also paid at $3 per bag. But, the total cost incurred by both are Z = ($6*number of Super-gro bags) + ($3*number of Crop-quick)

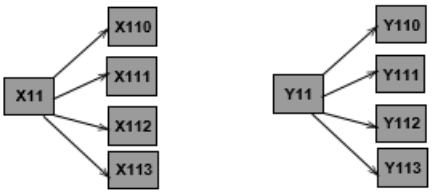

Figure 2. Representation of decision variable x11 and y11

LP Model Formulation

- Decision Variables:
  X1 = bags of Super-go
  X2 = bags of Crop-quick
- The Objective Function:
  Minimize Z = $6x1 + 3x2 + 4x3
  Where
  $6x1 = cost of bags of Super-go
  $3x2 = cost of bags of Crop-quick
- Model Constraints:
  2x1 + 4x2 >= 16lb(nitrogen constraint)
  4x1 + 3x2 >= 24lb(phosphate constraint)
  X1, X2 >= 0 (non negativity constraint)

$6x1 = cost of bags of Super-go
$3x2 = cost of bags of Crop-quick
- Model Constraints:
  2x1 + 4x2 >= 16lb(nitrogen constraint)
  4x1 + 3x2 >= 24lb(phosphate constraint)
  X1, X2 >= 0 (non negativity constraint)

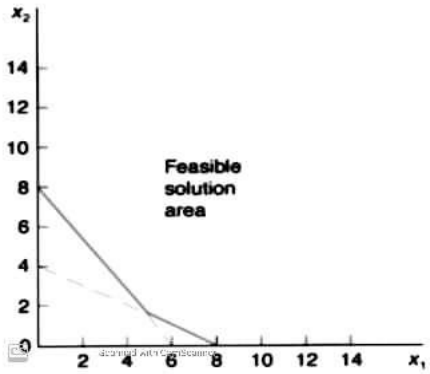

Figure 3. Graph of both Model Constraint

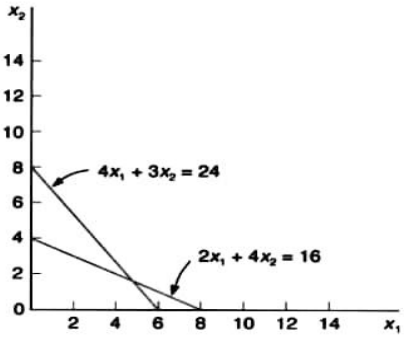

Figure 4. Feasible Solution Area

Minimize Z = $6x1 + $3x2 subject to: 2x1 + 4x2 16 4x2 + 3x2 24 x1, x2 0

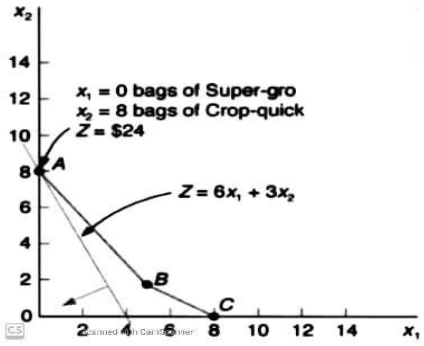

Figure 5. Minimization graph

SURPLUS VARIABLE

- A surplus variable is subtracted from a constraint to convert it to an equation (=).
- A surplus variable represents an excess above a constraint requirement level
- A surplus variable contributes nothing to the calculated value of the objective function.
- Subtracting surplus variables in the farmer problem constraints: 2x1 + 4x2 - s1 = 16 (nitrogen) 4x1 + 3x2 - s2 = 24 (phosphate)

Minimize Z = $6x1 + $3x2 + 0s1 + 0s2 subject to: 2x1 + 4x2 – s1 = 16 4x2 + 3x2 – s2 = 24 x1, x2, s1, s2 0

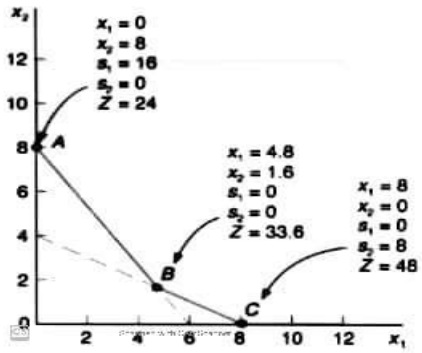

Figure 6.Graph of Fertilizer

The Linear Programming using Minisat+ is (pseudo code):

Inserting a list as: a=list()

Taking user input: n=input("Enter number of crops:")

Using a loop to enter the value one by one

for i in range (0,n):

    variable val = input("Enter cost crop ")

    taking input by appending val

Creating a list for nitrogen content

ni=list()

Using a loop to enter the value one by one

loop i in range n:

    variable val = input("Enter amount of nitrogen ")

taking input by appending val

Creating a list for phosphate content

ph=list()

Using a loop to enter the value one by one

loop i in range n:

    val = input("Enter amount of phosphet ")

    taking input by appending val

Total Nitrogen and Phosphate required:

totni = input("Enter total amount of nitrogen ")

totph = input("Enter total amount of phosphet ")

Taking a digit:

dig=7

Minimum required value:

str1="min:"

f=open("filewrite.opb","w"

Using loop to calculate the optimal solution

loop i in range n:

    when k=0

loop of k less than dig:

        co=power of 2 to k

str1=str1+plus+str of a of [i] multiply co+"multiply x"+str of I plus str of k

k= incrementing by 1

Using loop to calculate the equation as:

Z = $6x1 + $3x2 + 0s1 + 0s2

2x1 + 4x2 – s1 = 16

4x2 + 3x2 – s2 = 24

Where x1, x2, s1, s2 0

for i in range n:

k=0

loop k less than dig:

co=power of 2 to k

str1=str1+plus+str of nitrogen [i] multiply with co+"multiply x"+str of i+str of k

k=k+1

str1=str1 greater than or equal to str of total nitrogen

loop i in range n:

k=0

Loop k<=dig:

co=power of 2 to k

str1=str1+plus+str of phosphate [i] multiply with co+"multiply x"+str of i+str of k

k=k+1

str1=str1 greater than str of total phosphate

Loop: i in range n:

k=0

Loop k less than dig:

co=power of 2 to k

str1=str1+plus+str of co+"multiply x"+str of I +str of k

k=k+1

str1=str1 greater than 0

Ending the Code:

File write

File close

VI. OUTPUT

The output of the Minisat+ Solver When run, sends to standard error a number of different statistics about its execution. It will output to standard output either "SATISFIABLE" or "UNSATISFIABLE" depending on whether or not the expression is satisfiable or not.

If give a RESULT-OUTPUT-FILE, miniSAT will write text to the file. The first line will be "SAT" (if it is satisfiable) or "UNSAT" (if it is not). If it is SAT, the second line will be set of assignments to the Boolean variables that satisfies the expression.

In the code we have used demo input to test if the program is working and so the inputs taken are not related to practical use but it is ready to be implemented in real life optimizations of crop.

s SATISFIABLE

v -X114 -Y114 -X113 -Y113 -X112 -Y112 -X111 -Y111 -X110 -Y110 -X124 -Y124 -X123 -Y123 X122 Y122 -X121 -Y121 X120 Y120 -X134 -Y134 X133 -Y133 -X132 -Y132 X131 -Y131 -X130 -Y130 -X144 -Y144 X143 -Y143 -X142 -Y142 X141 -Y141 -X140 -Y140 -X154 -Y154 -X153 -Y153 X152 -Y152 X151 -Y151 X150 -Y150 -X164 -Y164 -X163 -Y163 -X162 -Y162 -X161 -Y161 -X160 -Y160 -X214 -Y214 -X213 -Y213 -X212 -Y212 X211 -Y211 -X210 -Y210 -X224 -Y224 -X223 -Y223 X222 Y222 X221 -Y221 -X220 -Y220 -X234 -Y234 X233 -Y233 -X232 -Y232 X231 -Y231 -X230 -Y230 -X244 -Y244 -X243 -Y243 X242 Y242 X241 -Y241 X240 -Y240 -X254 -Y254 -X253 -Y253 X252 -Y252 X251 -Y251 X250 -Y250 -X264 -Y264 -X263 -Y263 -X262 -Y262 X261 -Y261 X260 -Y260

The above output was got after implementing the equation with the demo variables and it is satisfiable which means the program is ready to be implemented in real life circumstances in optimizing the cost of crops.

VII. EXPERIMENTAL RESULTS

In this section, the crop optimization problem is experimented through the proposed PB-SAT based approach and the results are evaluated. The PB-constraints are created using python and solved through a well-known PB-SAT solver Minisat+ [29]. The experiments were conducted on an Intel Pentium 2.30 GHz system equipped with 4 GB of RAM, working on Linux environment. The timeout of SAT solver is set to 1000 seconds. The proposed approach is applied over 3 different test cases. Table III presents the potassium and nitrogen required for different test cases. The potassium and nitrogen requirement is mentioned for every fertilizer. The experiments are performed in every test case. The obtained results are shown in Table IV, V and VI. The potassium and nitrogen contents are taken the same as mentioned in the example. The total fertilizers cost required for two consecutive nutrients are presented in Table V for different test cases. The third column presented the CPU time (in seconds) required to execute the test cases. Table V demonstrates the "Super-Grow" assignments for every test case. Similarly, the "Crop-Quick" assignments for each interval of both the fertilizers are presented in Table VI. The observations coming out from the experimental results are presented in order:

The SAT solver works efficiently if the number of variables and constraints are less.

• The CPU time increased with the increase requirement of nutrients for any crop.

• The proposed approach is able to produce an efficient optimization for every test case.

VIII. CONCLUSION

This paper focuses on solving a crop optimization problem for a farm. Here, a pseudo-Boolean Satisfiability (PB-SAT) based technique is introduced to generate the fertilizers requirement for crops. The proposed approach expresses The optimization problem as a PB-SAT instances by formulating PB constraints. The PB-SAT solver Minisat+ is able to solve the PB-constraints efficiently and can generate an optimum fertilizer for the crops. The goal of the proposed technique is to minimize the total personnel cost while maintaining other fertilizer constraints provided. The proposed approach algorithm works efficiently on a varied set of inputs. Future work focuses on providing more flexibility to the optimization by applying additional constraints.

REFERENCES

[1] G. B. Dantzig, "Letter to the editorâ˘ A˘Ta comment on edie's â˘AIJtraffic delays at toll boothsâ˘ A˙I," Journal of the Operations Research Society of America, vol. 2, no. 3, pp. 339–341, 1954.

[2] J. Van den Bergh, J. Beliën, P. De Bruecker, E. Demeulemeester, and L. De Boeck, "Personnel scheduling: A literature review," European journal of operational research, vol. 226, no. 3, pp. 367–385, 2013.

[3] L. Wang and D.-Z. Zheng, "An effective hybrid optimization strategy for job-shop scheduling problems," Computers & Operations Research, vol. 28, no. 6, pp. 585–596, 2001.

[4] F. Pezzella, G. Morganti, and G. Ciaschetti, "A genetic algorithm for the flexible job-shop scheduling problem," Computers & Operations Research, vol. 35, no. 10, pp. 3202–3212, 2008.

[5] D. Moody, A. Bar-Noy, and G. Kendall, "Construction of initial neighborhoods for a course scheduling problem using tiling," in 2007 IEEE Symposium on Computational Intelligence in Scheduling. IEEE, 2007, pp. 187–191.

[6] Y. Yang, R. Paranjape, and L. Benedicenti, "An examination of mobile agents system evolution in the course scheduling problem," vol. 2, 06 2004, pp. 657 – 660 Vol.2.

[7] L. Wang, J. Cai, M. Li, and Z. Liu, "Flexible job shop scheduling problem using an improved ant colony optimization," Scientific Programming, vol. 2017, 2017.

[8] E. Burke, P. De Causmaecker, and G. V. Berghe, "A hybrid tabu search algorithm for the nurse rostering problem," in Asia-Pacific Conference on Simulated Evolution and Learning. Springer, 1998, pp. 187–194.

[9] M. Fırat and C. Hurkens, "An improved mip-based approach for a multiskill workforce scheduling problem," Journal of Scheduling, vol. 15, no. 3, pp. 363–380, 2012.

[10] M. A. O. Louly, "A goal programming model for staff scheduling at a telecommunications center," Journal of Mathematical Modelling and Algorithms in Operations Research, vol. 12, no. 2, pp. 167–178, 2013.

[11] S. A. Cook, "The complexity of theorem-proving procedures," in Proceedings of the Third Annual ACM Symposium on Theory of Computing, ser. STOC '71. New York, NY, USA: ACM, 1971, pp. 151–158. [Online]. Available: http://doi.acm.org/10.1145/800157.805047

[12] M. Moskewicz, C. Madigan, Y. Zhao, L. Zhang, and S. Malik, "Chaff: Engineering an efficient sat solver," vol. 2001, 02 2001, pp. 530 – 535.

[13] K. Pipatsrisawat and A. Darwiche, "A new clause learning scheme for efficient unsatisfiability proofs," in Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3, ser. AAAI'08. AAAI Press, 2008, pp. 1481–1484. [Online]. Available: http://dl.acm.org/citation.cfm?id=1620270.1620313

[14] J. Marques-Silva and K. A. Sakallah, "Grasp: A search algorithm for propositional satisfiability," IEEE Trans. Computers, vol. 48, pp. 506– 521, 1999.