

# Assigning final year students to project supervisors using Mixed Integer Linear Programming

Binaansim Stephen Npoandan, Opoku Rachel, Bismark  
Acheampong  
(Bachelor Of Science Mathematics)

Kwame Nkrumah University of Science and Technology, KNUST

**Supervisor: Dr. Peter Yerenkyi-Amoako**

December 16, 2021

# Outline

- Introduction
- Background
- Problem Statement
- Objectives
- Formulation of Our problem model
- Methodology-Mixed Integer Linear Programming
- Results and Analysis
- Conclusion and Recommendation
- References

# Introduction

- Final year students are required to apply their skills in solving problems as their project.
- Assigning them to supervisor contribute to effective project work carried out.
- This is tedious for large cohorts of students.
- We propose the use of Mixed Integer linear programming in finding optimal assignment as against traditional methods used

# Background

- Our focus is in the department of Mathematics, KNUST.
- There are 160 students and 18 lecturers in the department.
- The method used is by randomly assigning student in groups of two or three
- Areas of specializations for the finalist are;
  - Pure mathematics
  - Applied Mathematics
  - Mathematical Economics
  - Mathematical physics

# Problem Statement

The method for assigning student to supervisors in the department is either random assignment or try and error and;

- Turn to be tedious and time consuming for large cohorts of students
- Neglects student preference and lecturers workload
- Results in dissatisfaction assignments which are less optimal
- Less interest in project on the part of student.

# Problem statement - Cont

**Our problem then is:**

Finding optimal assignment of students to project supervisors  
using Mixed Integer Linear Programming.

# Objectives

- ① Modeling the problem as assignment LP problem.
- ② Find the optimal assignment using Mixed Integer programming
- ③ Automating the student supervisor allocation problem process.

# Student Supervisor Allocation, SSA problem

SSA problem entails;

- Assigning sets of students,  $S = \{1, 2, 3, \dots, s\}$  to  $L = \{1, 2, 3, \dots, l\}$
- Lecturer workload capacity collected in a vector known as Workload Vector
- Student preference over lecturer collected in the matrix,  $PM_{ij}$  also known as the cost matrix
- SSA can be formulated as a linear programming problem.



# Linear Programming, LP problems

SSA problems is an assignment problem and can be modelled as a linear programming problem;

- LP is optimization technique for allocating scarce resources to agents base on certain criterion of optimality.
- It consist of three components;
  - Decision variables
  - Objective function
  - Constraints.

The student supervisor allocation, SSA can be formulated as an LP problem as;

**Objective Function:**

$$\text{Minimize: } \sum_{i=1}^l \sum_{j=1}^s PM_{ij} X_{ij} \quad (1)$$

**Subject to:**

**1. Workload Constraint:**

$$\sum_{j=1}^s X_{ij} = W_i \text{ where } i \in L \quad (2)$$

**2. Students Constraint:**

$$\sum_{i=1}^l X_{ij} = 1 \text{ where } j \in S \quad (3)$$

**3. Nonnegativity Constraint:**

$$X_{ij} = 0 \text{ or } 1 \quad i \in L \text{ and } j \in S \quad (4)$$

# Parameters

- $l$  : number of lecturers in the department
- $s$  : number of students
- $W_i$  : number of students an  $i$ th lecturer can take
- $i$  : the  $i$ th index of students  $\{1, 2, 3, \dots, s\}$
- $j$  : the  $j$ th index of lecturers  $\{1, 2, 3, \dots, l\}$
- $PM_{ij}$  : The cost of assigning  $i$ th student to a  $j$ th lecturer.
- $X_{ij}$  : The decision variable.
- Decision variables are allowed to take the values;

$$X_{ij} = \begin{cases} 1, & \text{student } i \text{ is assigned to lecturer } j \\ 0, & \text{otherwise (No assignment is made)} \end{cases}$$

# Preference Matrix and Workload Vector

		Students				
		1	2	3	...	$s$
Lecturers	1	4	3	2		4
	2	3	1	4		8
	3	1	2	1		2
	...					
	$l$	5	7	6		1

Fig.1. A Students' Preference Matrix with  $l$  Lecturers and  $s$  Students

1	2	3	...	$l$
4	3	1	2	3

Fig.2. A Lecturer Workload Vector with  $l$  Lecturers

# Mixed Integer Linear Programming.

- An LP problem in which the decisions variables are restricted to integers is Integer Linear Programming, ILP.
- The type of ILP problems are;
  - Pure Integer Linear Programming
  - Mixed Integer Linear Programming, MILP
  - Binary Integer Integer Linear programming, BILP

# MILP model

An optimization problem in which the objective function is given to be a function

$$f : \mathbb{R}_+^{n \times p} \longrightarrow \mathbb{R}$$

and

$$X \subset \mathbb{R}_+^n \times \mathbb{Z}_+^p$$

serving the constraints,

$$X = \{(x, y) \in \mathbb{R}_+^n \times \mathbb{Z}_+^p : Ax + By \leq b\}$$

We define the objective function  $f$  as,

$$c^T x + h^T y$$

where  $c$  and  $h$  are vectors of length  $n$  and  $p$  respectively.

# Strategy to solve MILP problems

- Relax the solution space by replacing all the integer or binary variables with a continuous variable  $y$ , ( $0 \leq y \leq 1$ ). This model is known as Linear Relaxation, LR and it's solution is the same as the ordinary LP problem.
- Solve the Linear Relaxation problem and take note of the non-integer variables
- Impose special constraints that iteratively modify the solution space till it finally make optimum extreme point satisfy the the integer requirement that was initially imposed.

There are three main approach in finding the special constraints that help arrived at getting integer solutions;

- Branch and bound
- cutting plane
- Branch-and-cut

## **Branch-and-cut**

- Introduced in 1980s
- Makes solving large scale MILP problems possible
- It is a feature in most commercial solvers and open source solvers for MILP problems



# Implementation of MILP

- We implemented MILP using an open source cbc milp solver in Pulp library of the Python programming language, version 2.9.0 developed in February 12th 2015.
- The solver implements the theoretical descriptions given above.

# Advantages of the MILP over the Traditional Approach

The MILP approach Compared to the traditional approach;

- Easy to describe the Student Supervisor allocation, SSA problem
- Is less tedious for large student cohort since commercial solvers are used.
- Consumes less time

# Results and Interpretations

## Data

- 160 students to be assigned to 18 lecturers in the department of mathematics
- Randomly generated the Preference matrix,  $PM_{ij}$  in figure 1 below
- Used our discretion to generate the lecturer workload capacity

Lecturers/students	st1	st2	st3	st4	st5	st6	...	st160
lec1	2	9	18	2	17	16	...	14
lec2	13	6	11	18	13	2	...	10
lec3	9	1	11	7	5	15	...	5
lec4	10	18	15	8	1	1	...	13
lec5	5	9	5	7	16	16	...	12
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...	$\vdots$
lec18	1	9	4	6	3	5	...	6

Figure 1: Randomly generated Preference of 160 students in the department of mathematics, KNUST

lec1	lec2	lec3	lec4	lec5	lec6	lec7	lec8	lec9	lec10	lec11	lec12	lec13	lec14	lec15	lec16	lec17	lec18
12	6	9	12	10	10	9	8	9	18	4	8	9	9	4	6	8	9

Figure 2: Lecturer's Workload Vector,  $W_i$

Problem then is represented as:

Minimize

$$\sum_{i=1}^{18} \sum_{j=1}^{160} PM_{ij} X_{ij}$$

Subject to

$$\sum_{j=1}^{160} X_{ij} \leq W_i; \quad i \in Workload\_Vec,$$

$$\sum_{i=1}^{18} X_{ij} = 1; \quad j \in S$$

$$X_{ij} \geq 0; \quad X_{ij} \in 0, 1$$

# Expanded form:

Objective Function:

**Minimize**

$$\begin{array}{ccccccccccc} & 2X_{(1,1)} & + & 9X_{(1,2)} & + & 18X_{(1,3)} & + & \dots & + & 14X_{(1,160)} & + \\ & 13X_{(2,1)} & + & 6X_{(2,2)} & + & 11X_{(2,3)} & + & \dots & + & 10X_{(1,160)} & + \\ Z = & 9X_{(3,1)} & + & 1X_{(3,2)} & + & 11X_{(3,3)} & + & \dots & + & 5X_{(3,160)} & + \\ & \vdots & & \vdots & & \vdots & & \vdots & & \vdots & \\ & 1X_{(18,1)} & + & 9X_{(18,2)} & + & 4X_{(18,3)} & + & \dots & + & 6X_{(18,160)} & \end{array}$$

(This is made of  $18 \times 160$  which is 2880 decision variables.)

# 1st Constraint

**Subjected to:**

$$\begin{array}{cccccccccc} X_{(1,1)} & + & X_{(1,2)} & + & X_{(1,3)} & + & \dots & + & X_{(1,160)} & \leq & 12 \\ X_{(2,1)} & + & X_{(2,2)} & + & X_{(2,3)} & + & \dots & + & X_{(2,160)} & \leq & 6 \\ X_{(3,1)} & + & X_{(3,2)} & + & X_{(3,3)} & + & \dots & + & X_{(3,160)} & \leq & 9 \\ \vdots & & \vdots & & \vdots & & \vdots & & \vdots & & \vdots \\ X_{(18,1)} & + & X_{(18,2)} & + & X_{(18,3)} & + & \dots & + & X_{(18,160)} & \leq & 6 \end{array}$$

## 2nd Constraint

**And:**

$$\begin{array}{cccccccccc} X_{(1,1)} & + & X_{(2,1)} & + & X_{(3,1)} & + & \dots & + & X_{(18,1)} & = & 1 \\ X_{(1,2)} & + & X_{(2,2)} & + & X_{(3,2)} & + & \dots & + & X_{(18,2)} & = & 1 \\ X_{(1,3)} & + & X_{(2,3)} & + & X_{(3,3)} & + & \dots & + & X_{(18,3)} & = & 1 \\ \vdots & + & \vdots & + & \vdots & + & \vdots & + & \vdots & = & \vdots \\ X_{(1,160)} & + & X_{(2,160)} & + & X_{(3,160)} & + & \dots & + & X_{(18,160)} & = & 1 \end{array}$$



# Implementation

- **Solvers Used:** The mixed integer linear programming was implemented using version 2.9.0 cbc MILP solver in Pulp python package.
- **Computer System:** We carried out on an AMD A6-7310 APU core computer system with a processing speed of 2GHz and a RAM of 4 gigabyte running on a 64-bit windows 10 operating system.

# Results

After running the cbc milp solver on the objective function of 2880 variables subjected to 178 constraints;

- Objective values of 255.0
- It took 0.24 cpu seconds to generate the optimal assignment

# Optimal assignment

ASSIGN_lec1_st122: 1.0	ASSIGN_lec5_st96: 1.0	ASSIGN_lec10_st21: 1.0	ASSIGN_lec16_st79: 1.0
ASSIGN_lec1_st128: 1.0	ASSIGN_lec5_st98: 1.0	ASSIGN_lec10_st25: 1.0	ASSIGN_lec16_st92: 1.0
ASSIGN_lec1_st136: 1.0	ASSIGN_lec6_st100: 1.0	ASSIGN_lec10_st28: 1.0	ASSIGN_lec17_st110: 1.0
ASSIGN_lec1_st150: 1.0	ASSIGN_lec6_st105: 1.0	ASSIGN_lec10_st50: 1.0	ASSIGN_lec17_st121: 1.0
ASSIGN_lec1_st152: 1.0	ASSIGN_lec6_st108: 1.0	ASSIGN_lec10_st59: 1.0	ASSIGN_lec17_st157: 1.0
ASSIGN_lec1_st31: 1.0	ASSIGN_lec6_st109: 1.0	ASSIGN_lec10_st67: 1.0	ASSIGN_lec17_st159: 1.0
ASSIGN_lec1_st34: 1.0	ASSIGN_lec6_st115: 1.0	ASSIGN_lec10_st68: 1.0	ASSIGN_lec17_st29: 1.0
ASSIGN_lec1_st4: 1.0	ASSIGN_lec6_st125: 1.0	ASSIGN_lec10_st76: 1.0	ASSIGN_lec17_st42: 1.0
ASSIGN_lec1_st63: 1.0	ASSIGN_lec6_st23: 1.0	ASSIGN_lec10_st86: 1.0	ASSIGN_lec17_st49: 1.0
ASSIGN_lec1_st70: 1.0	ASSIGN_lec6_st37: 1.0	ASSIGN_lec11_st111: 1.0	ASSIGN_lec17_st95: 1.0
ASSIGN_lec1_st87: 1.0	ASSIGN_lec6_st44: 1.0	ASSIGN_lec11_st112: 1.0	ASSIGN_lec18_st1: 1.0
ASSIGN_lec1_st93: 1.0	ASSIGN_lec6_st7: 1.0	ASSIGN_lec11_st22: 1.0	ASSIGN_lec18_st116: 1.0
ASSIGN_lec2_st123: 1.0	ASSIGN_lec7_st131: 1.0	ASSIGN_lec11_st65: 1.0	ASSIGN_lec18_st141: 1.0
ASSIGN_lec2_st129: 1.0	ASSIGN_lec7_st133: 1.0	ASSIGN_lec12_st102: 1.0	ASSIGN_lec18_st24: 1.0
ASSIGN_lec2_st138: 1.0	ASSIGN_lec7_st135: 1.0	ASSIGN_lec12_st104: 1.0	ASSIGN_lec18_st36: 1.0
ASSIGN_lec2_st151: 1.0	ASSIGN_lec7_st142: 1.0	ASSIGN_lec12_st120: 1.0	ASSIGN_lec18_st64: 1.0
ASSIGN_lec2_st6: 1.0	ASSIGN_lec7_st17: 1.0	ASSIGN_lec12_st14: 1.0	ASSIGN_lec18_st66: 1.0
ASSIGN_lec2_st80: 1.0	ASSIGN_lec7_st33: 1.0	ASSIGN_lec12_st20: 1.0	ASSIGN_lec18_st84: 1.0
ASSIGN_lec3_st130: 1.0	ASSIGN_lec7_st41: 1.0	ASSIGN_lec12_st32: 1.0	ASSIGN_lec18_st85: 1.0
ASSIGN_lec3_st153: 1.0	ASSIGN_lec7_st52: 1.0	ASSIGN_lec12_st38: 1.0	
ASSIGN_lec3_st19: 1.0	ASSIGN_lec7_st91: 1.0	ASSIGN_lec12_st77: 1.0	
ASSIGN_lec3_st2: 1.0	ASSIGN_lec8_st103: 1.0	ASSIGN_lec13_st106: 1.0	
ASSIGN_lec3_st48: 1.0	ASSIGN_lec8_st124: 1.0	ASSIGN_lec13_st117: 1.0	
ASSIGN_lec3_st56: 1.0	ASSIGN_lec8_st147: 1.0	ASSIGN_lec13_st137: 1.0	
ASSIGN_lec3_st61: 1.0	ASSIGN_lec8_st156: 1.0	ASSIGN_lec13_st15: 1.0	
ASSIGN_lec3_st81: 1.0	ASSIGN_lec8_st158: 1.0	ASSIGN_lec13_st16: 1.0	
ASSIGN_lec3_st90: 1.0	ASSIGN_lec8_st35: 1.0	ASSIGN_lec13_st58: 1.0	
ASSIGN_lec4_st119: 1.0	ASSIGN_lec8_st69: 1.0	ASSIGN_lec13_st62: 1.0	
ASSIGN_lec4_st27: 1.0	ASSIGN_lec8_st99: 1.0	ASSIGN_lec13_st71: 1.0	
ASSIGN_lec4_st30: 1.0	ASSIGN_lec9_st12: 1.0	ASSIGN_lec13_st9: 1.0	
ASSIGN_lec4_st40: 1.0	ASSIGN_lec9_st145: 1.0	ASSIGN_lec14_st13: 1.0	
ASSIGN_lec4_st46: 1.0	ASSIGN_lec9_st146: 1.0	ASSIGN_lec14_st132: 1.0	
ASSIGN_lec4_st5: 1.0	ASSIGN_lec9_st154: 1.0	ASSIGN_lec14_st143: 1.0	
ASSIGN_lec4_st57: 1.0	ASSIGN_lec9_st155: 1.0	ASSIGN_lec14_st144: 1.0	

# Break down of the Assignments made

Lecturer (lec)	Student (st)	Supervision Capacity
lec1	st122, st128, st136, st150, st152, st31, st34, st4, st63, st70, st87, st93	12
lec2	st123, st129, st138, st151, st6, st80	6
lec3	st130, st153, st19, st2, st48, st56, st61, st81, st90	9
lec4	st119, st27, st30, st40, st46, st5, st57, st74, st75, st8, st82, st83	12
lec5	st107, st134, st139, st148, st160, st45, st88, st94, st96, st98	10
lec6	st100, st105, st108, st109, st115, st125, st23, st37, st44, st7	10
lec7	st131, st133, st135, st142, st17, st33, st41, st52, st91	9
lec8	st103, st124, st147, st156, st158, st35, st69, st99	8
lec9	st12, st145, st146, st154, st155, st26, st47, st60, st89	9
lec10	st10, st11, st113, st114, st118, st127, st140, st149, st18, st21, st25, st28, st50, st59, st67, st68, st76, st86	18
lec11	st111, st112, st22, st65	4
lec12	st102, st104, st120, st14, st20, st32, st38, st77	8
lec13	st106, st117, st137, st15, st16, st58, st62, st71, st9	9
lec14	st13, st132, st143, st144, st39, st43, st51, st53, st55	9
lec15	st101, st126, st54, st97	4
lec16	st3, st72, st73, st78, st79, st92	6
lec17	st110, st121, st157, st159, st29, st42, st49, st95	8
lec18	st1, st116, st141, st24, st36, st64, st66, st84, st85	9
		160

Figure 4: optimal allocation of students to supervisors

# Analysis

Choices	Number of Students	Percentage (%)
1 <sup>st</sup> Choice	99	61.875
2 <sup>nd</sup> Choice	35	21.875
3 <sup>rd</sup> Choice	22	13.75
4 <sup>th</sup> Choice	2	1.25
5 <sup>th</sup> Choice	0	
6 <sup>th</sup> Choice	2	1.25
Total	160	100
No. of Students who had 1 <sup>st</sup> – 3 <sup>rd</sup> choice lecturer	156	97.5
No. of Students who had 4 <sup>th</sup> – 6 <sup>th</sup> choice lecturer	4	2.5
Total	160	100

Figure 5: Percentage breakdown of the choices of students

# Pie chart representation

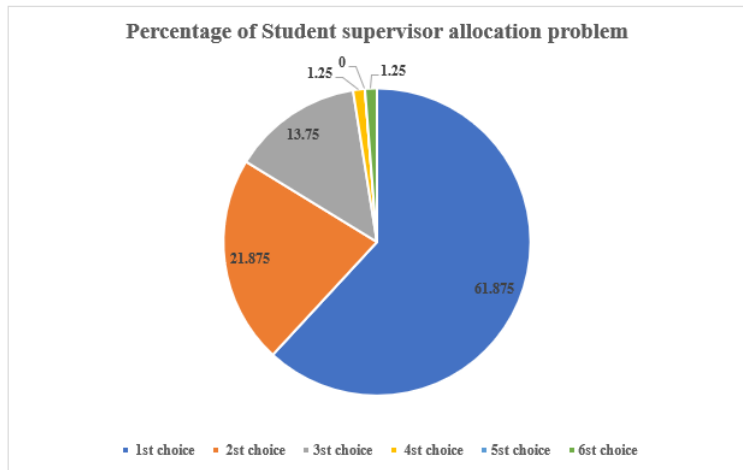


Figure 6: Pie chart representation of student preference

# Conclusion

In conclusion we;

- Modelled student supervisor allocation problem as an LP
- Arrived at optimal objective value of 255; (representing the minimum cost of dissatisfaction of the students and becomes the solution that gives the student the maximum satisfaction),
- Solution satisfying supervisors since no lecturer was assigned a number of students above their capacity.

# Recommendation

We agree that for large cohorts of students in final year our approach is most convenient and effective since;

- Consumes less time
- Arrives at a satisfactory solution to both students and lecturers



**Hence we recommend that;**

- ① This approach should be adopted by the department since it promises a more satisfying systematic approach to solving our allocation task
- ② A more realistic data be used since our data was randomly generated.






# Thank You!

# Referencing

-  J K SHARMA, "OPERATIONS RESEARCH THEORY AND APPLICATION", *PRESS TRINITY*, 2016, NEW DELHI - 110002, INDIA, 6th edition
-  DER-SAN CHEN, ROBERT G. BATSON and YU DANG, *John Wiley Sons Publication*, Applied Integer Linear Programming, 2010
-  Hamza O. Salami, and Esther Y. Mamman. "A Genetic Algorithm for Allocating Project Supervisors to Students", *I.J. Intelligent Systems and Applications*, 2016, 10

# Reference cont.

-  Victor Sanchez-Anguix, Rithin Chalumuri, Reyhan Aydoğan, Vicente Julian. "Allocation of Thesis Supervisor Using Genetic Algorithm", *Applied Soft Computing Journal*, 12 December 2018
-  Tim Mainhard, Roeland van der Rijst, Jan van Tartwijk, "A model for the supervisor-doctoral student relationship, *springerlink.com*, 2009, dio: 10.1007/s10734-009-9199-8.
-  A. Kwanashie, R. W. Irving, D. F. Manlove, C. T. Sng, "Profile-based optimal matchings in the Student/Project allocation problem" in combinatorial Algorithm, Springer International Publishing, 2014, pp. 213-225.