MATTER: International Journal of Science and Technology
ISSN 2454-5880

Prachapipat et al., 2018

Volume 3 Issue 3, pp. 253-270

Date of Publication: 1st February, 2018

DOI-https://dx.doi.org/10.20319/mijst.2018.33.253270

This paper can be cited as: Prachapipat, P, Leelertpanchai, A & Khancome, C. (2018). New Examination Timetabling Algorithm Using the Superstar Assignment Technique. MATTER: International Journal of Science and Technology, 3(3), 253-270.

NEW EXAMINATION TIMETABLING ALGORITHM USING THE SUPERSTAR ASSIGNMENT TECHNIQUE

Pornpun Prachapipat
Faculty of Science, Ramkhamhaeng University, Bangkok, Thailand
pornpun@ru.ac.th

Arkom Leelertpanchai
Faculty of Science, Ramkhamhaeng University, Bangkok, Thailand
arkoml776@gmail.com

Chouvalit Khancome
Faculty of Science, Ramkhamhaeng University, Bangkok, Thailand
chouvalit@hotmail.com

Abstract

In this paper, a new examination timetabling algorithm, SAT, is introduced. In order to solve the current problems that SAT algorithm can meet the requirement under the limited of all related resources and factors. It combined with rules, constraints, exceptions and priorities. This algorithm works in three steps: pre-processing, creating superstars and getting rid of superstars. SAT mechanisms sort all related parameters and factors, then determine stars and superstars of each related parameter. Each iteration of algorithm is trying to assign all superstars of all parameters as targets for processing. As well as, the process mechanism is run for putting the output into a suitable timeslot. To prove the algorithm implementation, a dataset from semester 1/2016 of Registration Centre, Ramkhamhaeng University has been selected as test subject. This dataset consists of 20/2 days/periods, 85,000 registered students, 1,325 subjects, 813,253 seats, and 11/76/22,582 buildings/rooms/seats per day. The proposed

model could be solved the current problems and shown details of subject such as the colour of answer sheet, room and so on. It could be done within less than two hours, meanwhile the current system took at least one month. For the future work, the large scale of algorithm is to be improved and developed into more dynamic version for a larger volume of data and handling more complicated constraints.

Keywords

Examination Timetabling Principle, Examination Timetabling Algorithm, Registration Problem, Superstar Assignment Technique

1. Introduction

Examination timetable is a classic problem, the solution of which includes finding an excellent algorithm to generate the target timetable under the limits of all related resources and factors. Traditionally, Genetic algorithm, Tabu search, and Evaluation theory are heuristic principles that have been employed to search for an answer to this problem.

Genetic algorithm, known as GA, was presented by John Holland (Goldberg, 1989) and was applied to examination timetabling. This algorithm is applied to provide the method to use all resources as each generation of gene to mix for each result mutation until getting the result in the last generation. There are many research articles applying this principle to solve the problem such as (Raghavjee & Pillay, 2010), (Raghavjee & Pillay, 2008), (Innet, 2013), and (Li, Lv, Mei, & Xu, 2010). Good reviews are also shown in (Ansari & Bojewar, 2014) and (Raghavjee & Pillay, 2011).

Presented by Fred Glover (Glover, 1986), Tabu Search algorithm employs the idea of searching in a suitable neighbourhood memory solution. This major process operates related resources and factors to put in a suitable timeslot of examination timetabling. If it is optimized, the process is finished or a new neighbourhood is set and operated until the result is optimized. Excellent examples are illustrated in (Santos, Ochi, & Souza, 2005), (Malik, Ayob, & Hamdan, 2010), and (Sabar, Ayob, & Kendall, 2009).

Evolutionary algorithm (EA) is based on Darwin's theory of evolution. The mechanism of this principle is to set the initial population, and iteratively refine the solution until certain termination criteria are met. The good review can be seen in (Cheong, Tan, & Veeravalli, 2007), (Mumford, 2007) and (Fernandes, Caldeira, Melicio, & Rosa, 1999).

Another important principle, Simulated Annealing (SA) was presented by Scott Kirkpatrick (Kirkpatrick, Gelatt, & Vecchi, 1983). Simulated Annealing algorithm was

applied with a new neighbourhood structure for the examination timetabling problem shown in (Liu, Zhang, & Leung, 2009). Besides, Ant colony optimization shown in Marco Dorigo (Dorigo, Maniezzo, & Colorni, 1996), as well as Harmony Search, was presented in (Al-Betar, Khader, & Nadi, 2010).

Recently, hybrid methodologies are chosen to present new alternative algorithms, such as (Cheong, Tan, & Veeravalli, 2007) —combining the structure of Heuristic Combinations in the Evolutionary Algorithm Hyper-Heuristic (Pillay, 2010). Harmony Search-based Hyper-heuristic (Anwar, Khader, Al-Betar, & Awadallah, 2013), and (Mandal & Kahar, 2015) are combination of a heuristic graph with the hill climbing search for solving complicated examination timetabling problem. Additionally, (Gupta, Narang, & Bansal, 2013) combines Active Rule, GA and a Hybrid Swarm-Based in (Fong, 2015). With more requirement of new examination timetabling algorithm, the Superstar Assignment Technique is emerged to solve this problem.

This article research presents new alternative algorithm called the SAT algorithm to solve the examination timetabling problem. The detail is presented in a general term of algorithm as shown in section 3. For the SAT's implementation, the examination timetabling of Ramkhamheang University is shown as the example results.

The reminders of this contribution are divided into 5 sections. Section 2 explains the preliminaries of the SAT. Then, section 3 shows the general term of algorithm. Section 4 illustrates the details of algorithm implementation, and section 5 is the experimental results and discussion. The conclusion and the future works are shown in section 6.

2. Preliminaries

To solve the examination timetable problem, it is required to find an excellent algorithm that can meet the requirement under the limited of all related resources and factors. As mentioned in (Kahar & Kendall, 2010), each university has a traditional way to solve its timetabling problem that meets their requirements under the limit of timeslots and other factors.

Moreover, the solution for examination timetabling problem also needs to take into account an optimization of timeslots, students, subjects, and examination rooms. Thus, there are many related factors such as the number of registered students in each subject, the objective or subjective of examination items, the rooms in each building, and so on to consider. In this section, all related variables for completing the Superstars Assignment

Technique will be shown. The following details show many variable factors to be used in the SAT algorithm.

- SUMMARY means a summary file(s) of all students that registered in all subjects.
- Objective is the number of all subjects that exam by objective examination item lists.
- Subjective is the number of all subjects that exam by subjective examination item lists.
- Target Time Slot refers to the number of periods per day and the number of the days to be used for each examination.
- Building means the number of buildings to be used for each examination.
- Room means the number of rooms in each building for each examination.
- Seat refers to the number of seats that are separated to even seats and odd seats in each room.
- Rule is the related rule to be used for controlling each examination timetabling.
- Constraint is the related constraint to be used for controlling each examination timetabling.
- Exception means related exceptions to be considered for each examination timetabling.
- Priority refers to the related priorities to be defined under each parameter.

3. General Term of Superstar Assignment Technique Algorithm

General term of SAT algorithm is divided to 3 steps: pre-processing, creating superstar, and trying to get rid of superstar(s). The pre-processing is the step for including all related factors to be defined as a major effect of requirement. Then every factor is classified for the superstar step. This step processes as a normal method of sorting data and selecting the first superstar preparation for getting rid of. In the last step, one by one superstar that are considered to create the solution. Then, each solution will be put into each suitable timeslot under the constraints, the limit of resources, all priorities and all exceptions. General term of SAT algorithm can be shown as below.

3.1 Step 1: Pre-Processing

- Include SUMMARY file(s)
- Include Objective and Subjective file(s)
- Include all related resources e.g., free Target Time Slots, Buildings, Rooms, Seats and etc.
- Include all related Rules, Constrains, Exceptions and Priorities

3.2 Step 2: Superstars

- Sort SUMMARY file(s) to the sorted SUMMARY file(s) and Get rid of the weakest number of data in file(s) using related rule(s);
- Determine superstars and stars of the registered student numbers
- Sort Objective and Subjective file(s) to the sorted Objective and Subjective files, and Get rid of the weakest number of data in file using related rule(s);
- Determine superstars and stars of Objective and Subjective items
- Determine superstars and stars of all resources considering the related constrains and priorities
- Create the blank target timeslot(s) preparing for the outputs

3.3 Step 3: Getting Rid of Superstars

- Read superstar of the sorted SUMMARY file(s), the sorted Objective and Subjective file(s), and all resources using related constrains and priorities
- Read all resources with their constrains and priorities and rules
- Determine the suitable position of superstar in the output timeslot under related rules and constrains
- Get rid of Superstar from the sorted SUMMARY file(s) and the sorted Objective and Subjective file(s), and/or all resources using related constrains and priorities if it is necessary
- Find and Determine a new superstar(s) of the sorted SUMMARY file(s) and the sorted Objective and Subjective file(s), and/or all resources using related constrains and priorities if it is necessary
- Repeat the first step (Read superstar) to find and determine a new superstar(s) until there are no any superstars.

4. Implementation

The details of SAT implementation are illustrated in two aspects: algorithm implementation and experiment implementation. All implementations employed the resources in Ramkhamhaeng University called RU.

The next sub-section shows the natural timetabling of RU, then all related parameters and the related rules from RU Registration Centre are shown. In the last sub-section, the algorithm implementation will be described.

4.1 Examination Timetabling of Ramkamhaeng University

According to information shown in (Ramkhamhaeng University, 2017), Ramkhamhaeng University (RU), the country's largest open university, was established in 1971 following a crisis in the quest for higher education, and was named after King Ramkhamhaeng the Great, renowned for inventing the Thai alphabet. Based on the principle of equality of opportunity to all regarding higher education, RU provides teaching and learning systems both on-campus and via distance learning. The university has approximately 85,000 registered students. In addition, the university has 1,019 academic staff for teaching and research as well as 3,025 administrative staff. At present, the university provides 5 examinations per academic year at the main campus (Huamark) and Bang Na campus. There are two final semester examinations, a summer session examination and two re-examinations. The problems of examination timetabling enlarge: the current system with the manual system takes a lot of time for timetabling with the big data, including the complicated constraints show as below.

- Any general education courses of the study plan have no the same examination period.
- Some big room (more than 100 seats), in each row provides 2 courses and allocate odd seats for one course and even seats for the other one.

In addition, these courses aren't the same colour of answer sheet. So each room provides 4 courses and 4 answer sheet colours. The steps of current system are shown as below.

The steps of legacy system flow (semi-automatic):

1. User prepare the paper of the available timeslots of every rooms of each period (room-no, number of row, number of seat per row, capacity)
2. User prints the report of sorted summary file (ascending sort: date, period, number of student, course-no)
3. Do step 4-5 until end of period of this examination.
4. User selects all courses of this period from the report, and takes the paper of the timeslot of this period.
5. Do step 5.1-5.6 until end of course of this period.

5.1 User takes the next course record of sorted summary file which is the maximum number of student

5.2 User selects the available timeslots of selected room by computing the number of allocated seats (the number of row with the number of seats) which support the number of student of this course and include above timetabling constraints.

5.3 User defines the colour of examination paper and the answer sheet.

5.4 On all of selected timeslots: user writes this course-no, the number of seats, the colour of answer sheet.

5.5 User reviews all of timeslot of this course

5.6 User writes timeslot data on the paper of timeslot form (data preparation of timeslot file)

6. User reviews all of the paper of timeslot form.

7. User inserts timeslot record into timeslot file.

This research proposed and designed new steps of system flow shown as below.

The steps of the new system (computerized system):

1. Create (one time) spreadsheet files of the available timeslots of every rooms of each period (room-no, number of row, number of seat per row, capacity) which are the same format.

2. Program processes 3-4 until end of period of this examination.

3. Program selects all courses of next period from sorted summary files (ascending sort: date, period, number of student, course-no)

4. Process step 4.1- 4.2 until end of course of this period.

4.1 Program reads a sorted summary record which maximum number of student

4.2 Program selects the available timeslots by computing the number of allocated seats (the number of row with the number of seats) which support the number of student of this course and include above timetabling constraints. Define the colour of the answer sheet.

- Program displays course-no and the colour of answer sheet on timeslot spreadsheet, insert a timeslot record (course-no, room-no, row-start, row-stop, the number of seats and the colour of answer sheet)

The proposed benefits of the new system provide the timetable processing in a short time with the timeslot data accuracy. Meanwhile, the computerized system provides the timeslot spreadsheets with the timetabling constraints in efficiency. Additionally, users can retrieve and verify selected timeslot data easier and the examination reports can be done immediately.

4.2 Configuration of Parameters

This sub-section shows how to use the previous parameters that are shown in section 2.

- SUMMARY = {S_1…S_s} where S_j is the unique subject.
- Objective/Subjective = {S_1:{o/s}, S_2:{o/s}, S_3:{o/s}, …. , S_s:{o/s}} where o is the objective subject and s is the subjective subject.
- Target Time Slots = {T_1:{p_1, p_2}, T_2 :{p_1, p_2}, T_3 :{p_1, p_2}, …, T_t :{p_1, p_2}} where T_i is the cell for putting a suitable subject, p_1 and p_2 are period 1 and period 2 respectively. (this research took 20 days where each day used two periods).
- Building = {B_1, B_2, B_3, ..., B_b} where B_k is the individual building to be used for the target examination.
- Room = {B_1:{Rm_1, Rm_2, Rm_3, …, Rm_{rm}}, B_2:{Rm_1, Rm_2, Rm_3, …, Rm_{rm}}, B_3:{Rm_1, Rm_2, Rm_3, …, Rm_{rm}}, …, B_b:{Rm_1, Rm_2, Rm_3, …, Rm_{rm}}} where each Rm_p is each individual room.
- Seat = {B1: {Rm_1:{rows:14, seats:18}, {Rm_2:{ rows:14, seats:18}, {Rm_3:{rows:14, seats:18},…,{Rm_r:{even:14, seats:18}}, …, B_b:{….}….},….} where rows:14 means Rm_1 has 14 rows and each row has 18 seats.
- Rule = {Ru_1…Ru_{ru}} where each Ru_s is each individual rule to be used for controlling an examination timetabling.
- Constraint = {C_1…C_c} where each C_t is the related constraints to be controlled the examination timetabling.
- Exception = {E_1…E_e} where each E_y is the related exceptions to be considered the examination timetabling.
- Priorities = {P:{L_1, L_2, L_3, …, L_l} where L_z is the level of priority to be defined to each related parameter.

In addition, the variable of superstar needs to be defined to every variable shown by the subscript. A symbol $_{st}$ means superstar. For example, if S_{st} is mentioned, then the superstar of Subject is determined, or Ru_{st} is shown, the superstar of Rule is declared for the first order parameter to be processed.

4.3 Algorithm Implementation

As shown in the general term of algorithm, the variables and all steps of the SAT algorithm that used the dataset from Ramkhamhaeng University are shown as below.

Step 1: Pre-Processing

- include $S_1 \ldots S_s$
- include subject that S_1:(o/s), S_2:(o/s), S_3:(o/s),…, S_s:(o/s)
- include $B_1 \ldots B_b$
- include B_1:{$Rm_1 \ldots Rm_{rm}$}, B2:{$Rm_1 \ldots Rm_{rm}$}, B3:{$Rm_1 \ldots Rm_{rm}$}, …, B_b:{$Rm_1 \ldots Rm_{rm}$},
- Include Seat:{B1:{Rm1:{row,seat},…, {Rmn:{row, seat}},…, B_n:{Rm_1:{row,seat}, …,{Rm_{rm}:{row,seat}}}
- include $Ru_1 \ldots Ru_{ru}$
- include $C_1 \ldots C_c$
- include $E_1 \ldots E_e$
- include priorities of any S, subjects, B, Rm, Ru, C and set a suitable $L_1 \ldots L_l$ to these variables

4.3.2 Step 2: Superstar

- Sort $S_1 \ldots S_s$,
- Sort subject that S_1:(o/s), S_2:(o/s), S_3:(o/s),…, S_s:(o/s) ,
- Sort $B_1 \ldots B_b$,
- Sort B_1:{$Rm_1 \ldots Rm_{rm}$}, B2:{$Rm_1 \ldots Rm_{rm}$}, B3:{$Rm_1 \ldots Rm_{rm}$}, …,B_b:{$Rm_1 \ldots Rm_{rm}$},
- Sort Seat:{B1:{Rm1:{row,seat},…, {Rmn:{row, seat}},…, B_n:{Rm_1:{row,seat},…, {Rm_{rm}:{row,seat}}} ,
- Sort $Ru_1 \ldots Ru_{ru}$,
- Sort $C_1 \ldots C_c$,
- Sort $E_1 \ldots E_e$,
- Sort priorities of any S, subjects, B, Rm, Ru, C and set a suitable $L_1 \ldots L_l$ to these variables
- Determine the first superstar considering to the first ordered of all parameters to S_{st}, B_{st}, Ru_{st}, C_{st}, E_{st}
- Create the output tables: T_1, T_2, T_3, …, T_t

4.3.3 Step 3: Getting Rid of Superstars

- Read S_{st} from the sorted file
- Read the related superstar of B_{st}, Ru_{st}, C_{st}, E_{st},
- Process and Create the suitable position of Buiding, Room, row, seat in the output table and mark that position and keep controlling all parameters using related rules by MS-Excel Macro and VB script
- Get rid of the first S_{st} and/or all superstars of all parameters if it is necessary

- Find the next superstar with the next prior and assign to superstar S_{st} and/or all parameters and set superstars to them
- Repeat first step (Read S_{st}) to the last step (Find the next superstar) until there is no superstar of S_{st}

5. Experimental Results and Discussions

The experiments were performed on a Dell Vostro 3400 notebook with Intel(R) CORE(TM) i5 CPU, M 560 @2.67 GHz, 4 GB of RAM, and running on Windows 7 Professional (32-bits) as an application machine. All programs were implemented using Microsoft Excel Version 2013. Dataset for testing used the dataset of semester 1/2016 from Ramkhamhaeng University. The dataset and the number of parameters inside the application are shown in the table 1 and 2 respectively.

Table 1: *Semester 1/2016 of RU Dataset*

Dataset	size
Student	85,000
Subject	1,325
Building	11
Room	76
Seats	**Total** = 813,253 seats Max/period = 22,582 seats Min/period = 18,245 seats
Objective/Subjective	200/1,105
Time(days)/periods	20/2
Campus	2

Table 2: *Parameters inside Application*

Parameters	**Total number**
Rules	101
Constrains	50
Exceptions	12
Input spreadsheets	76
Priority levels	5
Output spreadsheets	50

In the followings, some implementation of the input parameters represented in an application will be illustrated. Figure 1 below shows some details of SUMMARY file.

1	Course	Amount	Year	Month	Date	Period	Objective	Subjective
2	FIN2101	4260	59	10	20	A	080	000
3	POL2102	3170	59	10	20	A	000	003
4	THA1002	2338	59	10	20	A	120	000
5	LAW3004	2182	59	10	20	A	000	003
6	ANT3057	1974	59	10	20	A	100	000
7	SOC4077	1973	59	10	20	A	100	000
8	ENG2102	1163	59	10	20	A	000	003
9	CEC2201	413	59	10	20	A	020	016
10	CTH2105	339	59	10	20	A	000	005
11	PSY2002	306	59	10	20	A	100	000
12	PHY1105	299	59	10	20	A	080	000
13	PHY1101	298	59	10	20	A	080	000
14	POL4328	294	59	10	20	A	000	004
15	HRD2101	235	59	10	20	A	000	003
16	JPN2002	233	59	10	20	A	050	002
17	LAW4050	220	59	10	20	A	000	004
18	CHI3103	156	59	10	20	A	000	009
19	CTL3001	153	59	10	20	A	100	000
20	CEN4101	148	59	10	20	A	000	006
21	MCS3290	120	59	10	20	A	070	006
22	COS2103	80	59	10	20	A	000	005

Figure 1: *SUMMARY File*

Each objective and subjective spreadsheet consists of rules, constraints, and exceptions are shown in Figure 2.

1	Course	Objective	subjective	Hour : Minute
2	ACC1101	080	000	23
3	ACC1102	060	000	23
4	ACC1130	000	005	23
5	ACC2133	000	005	23
6	ACC2134	000	006	23
7	ACC2201	000	006	23
8	ACC2202	000	005	23
9	ACC3200	000	006	23
10	ACC3205	100	000	23

Figure 2: *Implementing Variables of Objective and Subjective Subject*

Some parts of each building that is combined with rules, constraints, exceptions, and priorities configurations are shown as Figure 3.

	Y	Grand Total			1,020	
		1	2	3	4	5
4	Total Seat	34	34	34	34	34
5	Used Seat	0	0	0	0	0
6	Remaing Seat	34	34	34	34	34
7	Odd Number	17	17	17	17	17
8	Seat - Odd No.	0	0	0	0	0
9	Remaing- Odd No.	17	17	17	17	17
10	Even No.	17	17	17	17	17
11	Seat - Even No.	0	0	0	0	0
12	Remaing- Even No.	17	17	17	17	17
13	Row / Process	Y	Y	Y	Y	Y

Figure 3: *Example of Buildings and Rooms Configuration*

Some parts of seats that are combined the rule configurations are shown in Figure 4.

	Seq No.	Room - No.	Row No.	Row Type	Column in table	Row Start
2	1	KLB-201	1	odd	B	14
3	2	KLB-201	2	even	C	14
4	3	KLB-201	3	odd	D	14
5	4	KLB-201	4	even	E	14
6	5	KLB-201	5	odd	F	14
7	6	KLB-201	6	even	G	14
8	7	KLB-201	7	odd	H	14
9	8	KLB-201	8	even	I	14
10	9	KLB-201	9	odd	J	14
11	10	KLB-201	10	even	K	14
12	11	KLB-201	11	odd	L	14
13	12	KLB-201	12	even	M	14
14	13	KLB-201	13	odd	N	14

Figure 4: *Implementing the Related Variables of Building, Rooms, Rows, Seats*

The application result for user is shown as Figure 5.

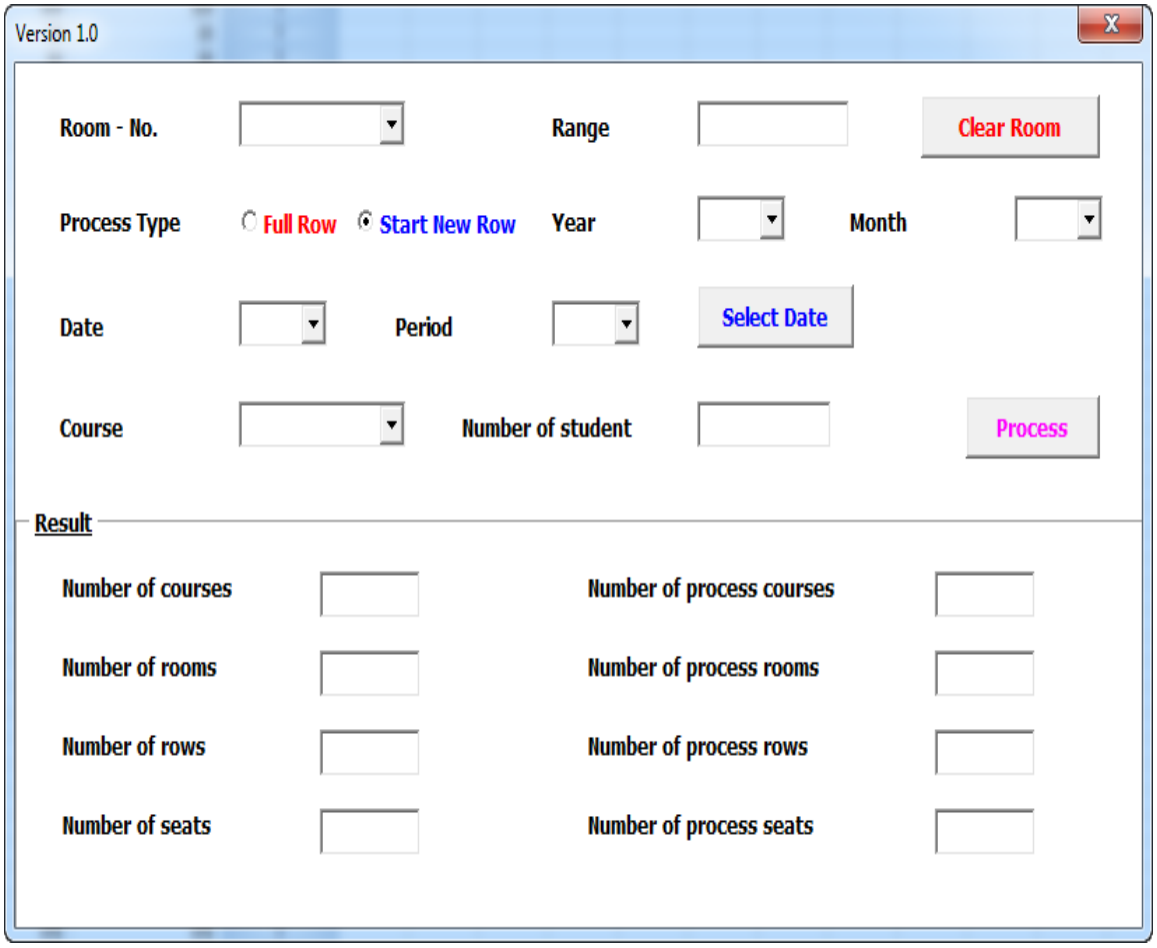

Figure 5: *Application for User*

The example result of a single room is shown in Figure 6.

Row / Process	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
1	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101
2	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101
3	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101
4	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101
5	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101
6	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101
7	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101
8	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101
9	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101
10	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101
11	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101
12	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101
13	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101
14	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101
15	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101
16	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101
17	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101
18	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101
19	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101
20	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101
21	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101
22	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101
23	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101
24	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101	3B1-FIN2101
25	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101	3A1-FIN2101

VKB-401

Figure 6: *Example of each output spreadsheet-room VKB-401 in one period per day*

As shown in each cell in Figure 6, the first row shows a number of seats, and each next column shows details of subject such as the colour of answer sheet, room, and so on.

For discussion from the experimental results, the examination timetabling period of Ramkhamheang University could be done within less than two hours, meanwhile the traditional examination timetabling took at least one moth.

6. Conclusions and Future Works

6.1 Conclusions

In this research article, new algorithm of examination timetable has been introduced. The new solution called SAT algorithm solves the problem by using the idea of superstar assignment technique. The algorithm steps consist of 1) pre-processing for inclusion of all related factors, rules and resources; 2) assigning the superstar and star to all parameters in step 1, then determining priorities called superstar, and 3) getting rid of the first superstar and shifting the next candidate superstar to superstar and repeating until there are no any further

superstars. Implementing this algorithm, the examination timetabling from semester 1/2016 of Ramkhamhaeng University has been selected for proving.

Research limitations are the dynamic rules, various constraints, limited rooms, exception, priority and so on. In addition, this paper proved the algorithm implementation with the only one dataset permission from semester 1/2016 of Registration center.

6.2 Future Works

For the future work, the large scale of algorithm is to be improved and developed into more dynamic version for a larger volume of data. Moreover, many more factors will be defined as targets. A new proposed model of examination timetabling problem will be created for handling more complicated constraints such as the number of examination staffs per room and more colors of answer sheet per row.

References

Al-Betar, M. A., Khader, A. T., & Nadi, F. (2010). Selection mechanisms in memory consideration for examination timetabling with harmony search. July 2010 GECCO'10: Proceeding of the 12th annual conference on Genetic and evolutionary computation, 12, 1203-1210. https://doi.org/10.1145/1830483.1830702

Ansari, A., & Bojewar, S. (2014, Nov). Genetic Algorithm to Generate the Automatic Time-Table – An Over View. International Journal on Recent and Innovation Trends in Computing and Communication (IJRITCC), 2 (11) , 3480-3483.

Anwar, K., Khader, A. T., Al-Betar, M. A., & Awadallah, M. A. (2013). Harmony Search-based Hyper-heuristic for examination timetabling, Signal Processing and its Applications (CSPA). 2013 IEEE 9th International Colloquium on, 9, 176–181. https://doi.org/10.1109/CSPA.2013.6530037

Cheong, C. Y., Tan, K. C., & Veeravalli, B. (2007). Solving the Exam Timetabling Problem via a Multi-Objective Evolutionary Algorithm– A More General Approach. Proceeding of the 2007 IEEE Symposium on Computational, Intelligence in Scheduling (CI-Sched 2007), 07, 165-172. https://doi.org/10.1109/SCIS.2007.367685

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant system:optimization by a colony of Cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 26, 29–41. https://doi.org/10.1109/3477.484436

Fernandes, C., Caldeira, J. P., Melicio, F.&Rosa, A. (1999). High school weekly timetabling by evolutionary algorithms. February 1999 SAC '99: Proceedings of the 1999 ACM symposium on applied computing, 344-350. https://doi.org/10.1145/298151.298379

Fong, C. W. (2015). Hishammuddin Asmuni, Barry McCollum, A Hybrid Swarm-Based Approach to University Timetabling. IEEE Transactions on Evolutionary Computation, 19, 870–884. https://doi.org/10.1109/TEVC.2015.2411741

Glover, F. (1986). Future Paths for Integer Programming and Links to Artificial Inlligence. Oxford: Elsevier Ltd.

Goldberg, DE. (1989). Genetic Algorithms in Search, Optimization and Machine Learning. 1st ed. New York: Addison-Wesley Longman Publishing Co.,Inc.

Gupta, A., Narang, B., & Bansal, R. (2013). Use of Active Rules and Genetic Algorithm to Generate the Automatic Time-Table. International Journal of Advances in Engineering Sciences, 3 (3), 40-43.

Innet, S. (2013). A Noval Approach of Genetic Algorithm for Solving Examination Timetabling Problems a case study of Thai Universities. 2013 13th International Symposium on Communications and Information Technologies (ISCIT), 13, 233-237. https://doi.org/10.1109/ISCIT.2013.6645855

Kahar, M.N.M., & Kendall, G. (2010). The examination timetabling problem at Universiti Malaysia Pahang: Comparison of a constructive heuristic with an existing software solution. European Journal of Operational Research, 207, 557–565. https://doi.org/10.1016/j.ejor.2010.04.011

Kirkpatrick, S., Gelatt, C., & Vecchi, M,. (1983, May 13). Optimization by Simulated Annealing. Science, 220 (4598) , 671–680. https://doi.org/10.1126/science.220.4598.671

Li, Xiaoping., Lv, Xiaoxing., Mei Wenbo., & Xu, Hu. (2010). Algorithm for solving Timetable questions based on Ga. The 3rd International Conference on Information Sciences and Interaction Sciences, 3, 18–21. https://doi.org/10.1109/ICICIS.2010.5534702

Liu, Y., Zhang, D., & Leung, S. C.H. (2009). A simulated annealing algorithm with a new neighborhood structure for the timetabling problem. June 2009 GEC '09:Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation, 9, 381-386. https://doi.org/10.1145/1543834.1543885

Malik, A. M. A., Ayob, M., & Hamdan, A. R. (2010). Stratified Random Sampling Techinique for Integrated Two-stage Multi-neighbourhood Tabu Search for Examination Timetabling Problem. 2010 10th International Conference on Intelligent Systems Design and Applications, 1326-1331 . https://doi.org/10.1109/ISDA.2010.5687093

Mandal, A. K., & Kahar, M. N. (2015). Combination of graph heuristic with hill climbing Search for solving capacitated examination timetabling problem. the 4th International Conference on Software Engineering and Computer Systems (ICSECS) 2015, 4, 118-123. https://doi.org/10.1109/ICSECS.2015.7333095

Mumford, C. L. (2007). An Order Based Evolutionary Approach to Dual Objective Examination Timetabling– A More General Approach, Proceedings of the 2007 IEEE Symposium on Computational, Intelligence in Scheduling (CI-Sched 2007), 07, 176-186. https://doi.org/10.1109/SCIS.2007.367687

Pillay, N. (2010). An empirical study into the structure of heuristic combinations in an evolutionary algorithm hyper-heuristic for the examination timetabling problem. October 2010 SAICSIT '10: Proceedings of the 2010 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists, 10, 251-257. https://doi.org/10.1145/1899503.1899531

Raghavjee, R., & Pillay, N. (2008). An application of genetic algorithms to the school timetabling problem. SAICSIT '08: Proceedings of the 2008 annual research conference of the South African Institute of Computer Scientists and Information Technologists on IT research in developing countries: riding the wave of technology, 8, 193-199. https://doi.org/10.1145/1456659.1456682

Raghavjee, R., & Pillay, N. (2010). An informed genetic algorithm for the high school timetabling problem. October 2010 SAICSIT '10: Proceedings of the 2010 Annual Research Conference of the South African Institute of Computer Scientists and Informationn Technologists, 10, 408-412. https://doi.org/10.1145/1899503.1899555

Raghavjee, R., & Pillay, N. (2011). The effect of construction heuristics on the performance of a genetic algorithm for the school timetabling problem. SAICSIT '11: Proceedings of the South African Institute of Computer Scientists and Information Technologists Conference on Knowledge, Innovation and Leadership in a Diverse, Multidisciplinary Environment, 11, 187-194 . https://doi.org/10.1145/2072221.2072243

Tito, Amranes. (2017, July 8). RU Background. *Institute of International Studies (IIS-RU) Ramkhamhaeng University*. Retrieved from http://www.ru.ac.th/index.php/ru-background

Sabar, N. R., Ayob, M., & Kendall, G. (2009). Tabu exponential Monte-Carlo with counter heuristic for examination timetabling. 2009 IEEE Symposium on Computational Intelligence in Scheduling, 90-94. https://doi.org/10.1109/SCIS.2009.4927020

Santos, H. G., Ochi, L. S., & Souza, M. J.F. (2005). A Tabu search heuristic with efficient diversification strategies for the class/teacher timetabling problem. December 2005 Journal of Experimental Algorithmics, 10, 1-16. https://doi.org/10.1145/1064546.1180621