Construction of Initial Solution Population for Curriculum-Based Course Timetabling using Combination of Graph Heuristics

Juliana Wahid[1] and Naimah Mohd Hussin[2]
[1]*School of Computing, Universiti Utara Malaysia, 060101 UUM, Sintok, Kedah.*
[2]*Faculty of Computer Science and Mathematical, Universiti Teknologi MARA (Perlis), 02600 Arau, Perlis.*
w.juliana@uum.edu.my

***Abstract*—The construction of population of initial solution is a crucial task in population-based metaheuristic approach for solving curriculum-based university course timetabling problem because it can affect the convergence speed and also the quality of the final solution. This paper presents an exploration on combination of graph heuristics in construction approach in curriculum based course timetabling problem to produce a population of initial solutions. The graph heuristics were set as single and combination of two heuristics. In addition, several ways of assigning courses into room and timeslot are implemented. All settings of heuristics are then tested on the same curriculum based course timetabling problem instances and are compared with each other in terms of number of population produced. The result shows that combination of largest degree followed by saturation degree heuristic produce the highest number of population of initial solutions. The results from this study can be used in the improvement phase of algorithm that uses population of initial solutions.**

***Index Terms*—Construction Phase; Curriculum-Based Timetabling; Graph Heuristics; Population-Based Metaheuristic.**

I. Introduction

The curriculum-based university course timetabling (CBCTT) is a process that allocate lectures to specific rooms and timeslots based on the university curricula. The assigning courses to rooms and periods are subjected to satisfy several constraints so that a feasible timetable can be produced. Similar to other combinatorial optimization problems, the solution for CBCTT typically involved two stages, i.e. the construction and the improvement phases[1]. In the first phase, the approach begins with an empty timetable and progressively inserting a lecture into the timetable one by one in iterative manner. There will be no hard constraints (feasible) in the initial timetable produced but mostly may have many soft constraints violations. The improvement algorithm begins with an initial timetable and then tries to gradually improve it, with respect to the objective function. In each step of the improvement algorithm, some elements of the timetable may be changed hoping to achieve a better timetable.

The construction of population of initial solution is a prerequisite in a population-based metaheuristic implementation. To produce a population of initial solution require algorithm that can produce multiple feasible solutions and these solutions must be diverse. This process is a crucial task because it can affect the convergence speed and also the quality of the final solution [2].

This paper investigates the construction phase approach in which several graph heuristics are combined to generate a population of initial solutions. This study able to produce a set of initial solution, therefore it is able to contribute to the improvement phase of approach that uses population of initial solutions such as ant colony optimization (ACO)[3], genetic algorithm (GA)[4], and harmony search algorithm (HSA)[5]. The approach in this study also shows that a feasible timetable can be found for numerous data set problems.

This paper is arranged as follows; section 2 provides the literature review on the description of CBCTT and similar works that use same domain of problem. Section 3 discusses the proposed approach, while section 4 presents the computational results and analysis. The conclusion and future direction are stated in final section.

II. Literature Review

A. Curriculum-Based Course Timetabling (CBCTT)

The CBCTT problems consists of the allocation of a set of lectures (derived from courses) to a set of rooms and periods. The allocation is establish base on a set of constraints and designated on a weekly basis. The entities involved in this problem are days (d) - number of teaching days in a week (usually 5 or 6), timeslots (ts) – number of time gap in a day (equal for all days), periods (P) = d X ts - a pair consisted of a day and a timeslot, courses and teachers - each course is comprised of the number of lectures to be scheduled and each lecture is related to a teacher, rooms - each room has a capacity (number of available seats), and curricula - a curricula is a group of courses such that any pair of courses in the group have students in common. The conflicts between courses are produced based on the curricula.

Timetable is feasible if all the hard constraints are not violated but may contains the violations of the soft constraints. The violation of the soft constraints is presented by number of penalty cost. Therefore, the aim of the CBCTT problem is to minimize the penalty cost of the soft constraint violations in a feasible timetable.

This study adopts the hard and soft constraints used by [6]. The hard constraints are lectures, room occupancy, conflicts and availability, while the soft constraints are room capacity, minimum working days, curriculum compactness and room stability. The detail of each constraint is described in Table 1.

Table 1
Description of Constraints

Type	Constraints	Description
Hard constraints	Lectures	All lectures must be scheduled to a different period and a room
	Room occupancy	Two lectures cannot be assigned to the same room and the same period
	Conflicts	Lectures in the same curriculum or taught by the same teacher must be assigned to separate periods
	Availability	If the teacher of a course is not available at a certain period, then no lectures of the course can be allocated to that period
Soft constraints	Room capacity	The quantity of students for each lecture must be fewer or equivalent to the volume of the rooms
	Minimum working days	The lectures of each course should be span thru a given number of days
	Curriculum compactness	Lectures of courses of the same curriculum should be in contiguous periods
	Room stability	All lectures of a specific course are supposed to be allocated to the same room

The quality of solution is calculated as the total penalties of the soft constraints: room capacity + minimum working days + curriculum compactness + room stability.

B. Similar Works

This section reviews other methods in the literature that use the same CBCTT benchmark data sets and graph heuristics for their initial solution construction.

[7] generates a feasible initial solution satisfying all the hard constraints by using two phase algorithm. Starting from an empty timetable, the first phase iteratively scheduled an event that has the highest conflict until the hard constraints are satisfied. If a feasible solution is found, the algorithm stops. Otherwise, phase two is executed. In the second phase, neighborhood N1: move a randomly selected lecture and/or N2: swap two lectures at random are applied. N1 is applied at a maximum of 500 iterations. If a feasible solution is met, then the algorithm stops. Otherwise the algorithm continues by applying an N2 neighborhood structure at a maximum of 500 iterations. This approach has been tested on 21 datasets and successfully achieved a feasible solution for all datasets. This approach is also used by [8].

[9] used hybrid construction heuristic that consists of three phases, i.e. Phase 1: largest degree heuristic, Phase 2: neighborhood search, and Phase 3: tabu search. In Phase 1, the courses with the highest number of conflicts are firstly scheduled to specific room and timeslot while fulfill the hard constraints. In certain events, a feasible timetable is found by simply applying Phase 1, in which Phase 2 and Phase 3 are not needed. However, feasibility from Phase 1 is not assured, and Phase 2 and Phase 3 are then executed until feasibility has been attained. In phase 2, two neighborhood moves are used such as Nbs1: Select a course at random and move to another random feasible timeslot-room, and Nbs2: Select two courses randomly and swap their timeslots and rooms while ensuring feasibility is maintained. The process stops if there is no improvement on the current timetable after 20 iterations. In Phase 3, tabu search algorithm is used to be scheduled the courses into the timetable. To ensure feasibility, the authors had implemented three phases approach where the second and third phases are normally considered as improvement phase. Sometimes heuristics algorithms are not able to generate feasible solution in the initial phase because with complex data set, some lectures may not have valid slot or time period. This approach is also used by [10].

III. The Proposed Approach

The proposed approach is shown in Figure 1. The first step in the approach is to determine the sequential order of courses/lectures to be schedule using the combination of graph heuristics. From six different graph heuristics described by [11], this study investigates only three type of heuristics. These heuristics have been chosen because they are the most widely applied graph heuristics and have produced feasible initial solutions for all data instances of CBCTT [9][7][12]. The graph heuristics are largest degree (events that have a large number of conflicts with other events are scheduled earlier), weighted degree (events that have large number of students in the conflict are scheduled earlier), saturation degree (events that have the smallest number of free valid periods are assigned earlier).

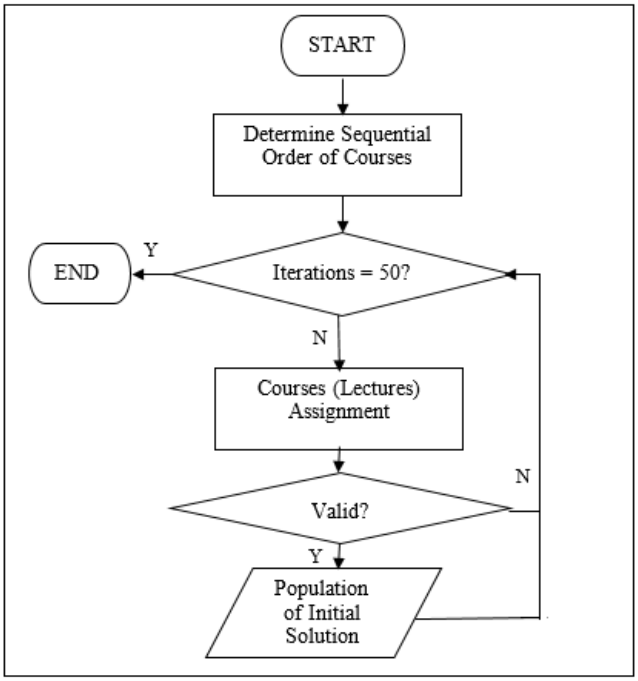

Figure 1: Approach for Population of Solution Construction

The courses/lectures were ordered using single heuristic, and a combination of two heuristics. The ordering method is identified by the following label of combination(s): L (largest degree), W (weighted degree), S (saturation degree), LS

(largest degree with saturation degree, WS (weighted degree with saturation degree), SL (saturation degree with largest degree), and SW (saturation degree with weighted degree).

The weighted degree is a heuristic that orders the events by the descending number of students involved in conflicts. This heuristic already contains the largest degree (descending number of conflicts) heuristic, therefore there is no combination between largest degree and weighted degree.

In the second step, each of the courses/lectures which is previously arranged based on the heuristics setting will be randomly and iteratively allocated to valid empty slots while satisfying all the hard constraints. If a lecture unable to be allocated to any slots due to no more valid empty slots, it will be added into the unscheduled lectures record. The unscheduled lectures record will be assigned later to the timetable using several methods that executed in a sequence.

After all the lectures are assigned, the timetable will be validated in terms of penalties of hard constraints violation. If the penalty is zero means that the timetable is feasible. The algorithm keeps the feasible timetable and begins another courses (lectures) assignment step with different random seeds. Else if the penalty of the hard constraints violation is not zero, the existing timetable will be removed and another courses (lectures) assignment step will be carried out with a different random seed.

The step of courses (lectures) assignment will be repeated for 50 iterations so that a maximum of 50 feasible timetables can be constructed.

IV. Experimental Results and Analysis

The proposed approach was implemented using C++ language and tested using computer with an Intel Core i7 with 3.4 GHz processor. The experiment was carried out using 21 data instances available at http://tabu.diegm.uniud.it/ctt website. All seven graph heuristic settings (L, W, S, LS, WS, SL, SW) were performed for each data instance, without imposing a time limit as a stopping condition. For each data instance, the total number of feasible solutions of over 50 iterations will be produced.

The results of individual graph heuristic of L, W and S and combination of graph heuristics of LS, WS, SL and SW are shown in Table 2, 3, and 4 respectively. TOT represents the total number of feasible initial solutions that produced over 50 iterations, while MIN and MAX denote as the minimum cost and maximum cost of soft constraints violation that is produced.

In Table 2, 3 and 4, for all settings, the problem instance of comp05 is not able to produce at least 10 percent of initial solutions over 50 iterations. This is probably because of the complexity of the problem instance. Even though the problem instances of comp12 is not able to produce the maximum number of initial solutions, at least it produces a number of initial solutions that is near to the 50 solutions. Therefore, for the purpose of determining the best setting, the number of feasible initial solution for comp05 in each setting will be compared. As shown in Table 3, comp05 in LS consists of the highest number of feasible initial solutions, i.e. 4 compared to other settings. Based on this, it can be concluded that the graph heuristic LS is the best setting in producing population of initial solution.

Table 2
Results of Initial Solution Using Single Heuristic

Problem	L			W			S		
Instances	TOT	MIN	MAX	TOT	MIN	MAX	TOT	MIN	MAX
comp01	50	346	646	50	365	900	50	272	525
comp02	50	781	1409	50	757	1586	50	728	1482
comp03	49	682	1193	50	705	1277	50	724	1157
comp04	50	708	843	50	696	987	50	702	871
comp05	**2**	**1625**	**2027**	**3**	**1405**	**1769**	**2**	**1526**	**1999**
comp06	50	957	1276	50	952	1595	50	961	1360
comp07	50	1080	1297	50	1051	1401	50	1092	1263
comp08	50	780	92	50	773	962	50	787	919
comp09	50	839	1033	50	849	1041	50	852	1032
comp10	50	916	1071	50	884	1126	50	936	1100
comp11	50	203	308	50	288	712	50	220	329
comp12	47	1552	2057	50	1610	2281	46	1547	1909
comp13	50	810	973	50	802	984	50	816	963
comp14	50	710	913	50	733	901	50	723	889
comp15	49	682	1193	50	705	1277	50	724	1157
comp16	50	937	1127	50	968	1354	50	924	1150
comp17	48	908	1315	48	887	1323	48	902	1077
comp18	50	608	782	50	581	795	50	636	764
comp19	50	707	1107	50	715	1246	50	706	1158
comp20	50	1011	1280	50	1260	2495	50	1040	1437
comp21	50	942	1285	50	952	1330	50	918	1452

Table 3
Results of Initial Solution Using Combination Heuristic (LS and WS)

Problem	LS			WS		
Instances	TOT	MIN	MAX	TOT	MIN	MAX
comp01	50	330	569	50	326	690
comp02	50	769	1345	50	762	1424
comp03	50	702	881	50	701	1209
comp04	50	694	831	50	705	934
comp05	4	1466	1890	2	1313	1750
comp06	50	947	1448	50	964	1473
comp07	50	1043	1310	50	1077	1255
comp08	50	790	929	50	793	934
comp09	50	847	1064	50	854	1040
comp10	50	898	1074	50	898	1112
comp11	50	230	312	50	264	660
comp12	50	1498	2044	50	1504	2236
comp13	50	793	969	50	803	943
comp14	50	745	903	50	724	904
comp15	50	702	881	50	701	1209
comp16	50	949	1117	50	936	1328
comp17	48	902	1270	43	901	1288
comp18	50	583	773	50	636	775
comp19	50	637	1225	50	675	935
comp20	50	1042	1282	50	1158	2177
comp21	50	928	1260	50	913	1310

Table 4
Results of Initial Solution Using Combination Heuristic (SL and SW)

Problem	SL			SW		
Instances	TOT	MIN	MAX	TOT	MIN	MAX
comp01	50	323	489	50	366	559
comp02	50	747	1356	50	779	1377
comp03	50	715	1129	50	690	1125
comp04	50	692	862	50	717	872
comp05	**3**	**1594**	**2098**	**1**	**1296**	**-**
comp06	50	982	1273	50	953	1368
comp07	50	1063	1270	50	1059	1243
comp08	50	788	944	50	780	948
comp09	50	849	999	50	811	1067
comp10	50	920	1104	50	943	1099
comp11	50	215	327	50	221	339
comp12	49	1542	1919	49	1426	1897
comp13	50	818	990	50	793	960
comp14	50	720	898	50	732	888
comp15	50	715	1129	50	690	1125
comp16	50	965	1163	50	942	1129
comp17	48	910	1198	47	867	1098
comp18	50	627	776	50	604	796
comp19	50	668	1047	50	670	1025
comp20	50	1036	1552	50	1009	1560
comp21	50	906	1177	50	956	1470

To verify that the graph heuristic LS is the best setting of graph heuristic in producing feasible initial solutions, this setting will be applied to other problem instances that are also available at http://tabu.diegm.uniud.it/ctt/index.php such as:

i. DDS-2008 - Instances proposed by De Cesco, Di Gaspero, & Schaerf in 2008 (7 data intances: DDS1, DDS2, DDS3, DDS4, DDS5, DDS6, DDS7)
ii. Test - Test instances (5 data intances: test1, test2, test3, test4, toy)
iii. Erlangen* (*Very large instances, provided by Moritz Muehlenthaler)- Instances from University of Erlangen (Germany) (6 data instances: Erlangen2011_2, Erlangen2012_1, Erlangen2012_2, Erlangen2013_1, Erlangen2013_2, Erlangen2014_1)
iv. UniUD - New instances from University of Udine (Italy) (9 data instances: Udine1, Udine2, Udine3, Udine4, Udine5, Udine6, Udine7, Udine8, Udine9)
v. EasyAcademy* - Instances from various Italian Universities (12 data instances: EA01, EA02, EA03, EA04, EA05, EA06, EA07, EA08, EA09, EA10, EA11, EA12)

Table 5 shows the results of applying graph heuristic of LS for the above problem instances. The use of graph heuristic of LS can produce maximum population of feasible initial solution for most of the problem instances, while at least producing several initial solutions for complex problem instances such as for DDS1, EA01, EA02, and EA06.

Table 5
Graph Heuristic of LS in Producing Feasible Initial Solutions (over 50 Iterations) for other Problem Instances

Problem Instances	TOTAL	MIN	MAX
DDS1	3	10411	11187
DDS2	50	176	251
DDS3	50	147	224
DDS4	50	3458	5109
DDS5	50	844	1021
DDS6	50	838	1319
DDS7	50	623	770
test1	38	1200	1423
test2	50	448	1118
test3	50	558	884
test4	50	631	857
Erlangen2011_2	20	46066	49093
Erlangen2012_1	50	55332	57837
Erlangen2012_2	50	65867	72503
Erlangen2013_1	50	50990	53274
Erlangen2013_2	50	57582	61366
Erlangen2014_1	50	47431	53655
Udine1	50	885	1412
Udine2	50	913	1129
Udine3	50	881	1320
Udine4	50	615	772
Udine5	50	831	1022
Udine6	50	621	1130
Udine7	50	688	1336
Udine8	50	968	1409
Udine9	50	759	1098
EA01	8	808	1062
EA02	5	269	561
EA03	50	1060	1601
EA04	50	782	935
EA05	50	413	968
EA06	4	265	1247
EA07	48	890	1179
EA08	50	483	586
EA09	49	682	870
EA10	50	960	1295
EA11	50	200	332
EA12	50	226	312

A. Comparison with Previous Similar Works

The obtained results (MIN value) of LS graph heuristic are compared to those of previous similar works described in the above section. However, most of the reviewed similar works did not highlight the initial solution output, as their focus was more on the output of the improvement phase. Only two works, i.e. [9] and [10] demonstrated their initial solution output. These two works show the same result of initial solution (as the authors in these works are the same person). In these works, hybrid construction heuristics which consisted of largest degree heuristic, neighborhood search, and tabu search are used for initial solution construction.

Table 6 shows the comparison between the initial solutions of the proposed algorithm with the initial solution produced by these two works. The focus of these two works is on obtaining the optimal result of the CBCTT, while the focus of this paper is to produce population of initial solution. Therefore, the comparisons are only meant to show the ability of the proposed method to find population of initial solutions to the CBCTT.

Table 6
Comparison of Initial Solution between Proposed Method with Previous Method

	Initial Solution	
Problem Instances	[9] [10]	Proposed method
comp01	1869	330
comp02	6776	769
comp03	6041	702
comp04	4429	694
comp05	7513	1466
comp06	4310	947
comp07	3119	1043
comp08	3007	790
comp09	4537	847
comp10	2479	898
comp11	1212	230
comp12	3155	1498
comp13	4828	793
comp14	3254	745
comp15	5717	702
comp16	4888	949
comp17	3808	902
comp18	1495	583
comp19	4609	637
comp20	5852	1042
comp21	4459	928

V. CONCLUSION

The aim of this paper was to inspect the implementation of graph heuristics combinations in CBCTT for generating population of initial solutions. Result demonstrates that the use of largest degree followed by saturation degree, created maximum number of population instead of the use of single graph heuristics. The result of this study can be applied in the second phase of solving CBCTT that is the implementation phase, so that the solution (timetable) will be optimize to the lowest number of soft constraints, i.e. near to optimal solution.

ACKNOWLEDGMENT

We are grateful for the KPT scholarship to Author 1.

REFERENCES

[1] S. Petrovic, "Towards the Benchmarks for Scheduling," in *The International Conference on Automated Planning and Scheduling, ICAPS07*, 2007.

[2] S. Rahnamayan, H. R. Tizhoosh, and M. M. a. Salama, "A novel population initialization method for accelerating evolutionary algorithms," *Comput. Math. with Appl.*, vol. 53, no. 10, pp. 1605–1614, 2007.

[3] K. Socha, K. Joshua, and S. Michael, "A max-min ant system for the university course timetabling problem," *Proc. 3rd Int. Work. Ant Algorithms (ANTS 2002).Lecture Notes Comput. Sci. Vol. 2463. Springer-Verlag, Berlin, pp. 1-13*, 2002.

[4] R. Lewis and B. Paechter, "Application of the Grouping Genetic Algorithm to University Course Timetabling," in *Evolutionary Computation in Combinatorial Optimization*, vol. 3448, G. Raidl and J. Gottlieb, Eds. Springer Berlin / Heidelberg, 2005, pp. 144–153.

[5] M. A. Al-Betar and A. T. Khader, "A harmony search algorithm for university course timetabling," *Ann. Oper. Res.*, vol. 194, no. 1, pp. 1–29, 2010.

[6] L. Di Gaspero, B. McCollum, and A. Schaerf, "The Second International Timetabling Competition (ITC-2007): Curriculum-based course timetabling (track 3)," School of Electronics, Electrical Engineering and Computer Science, Queens University, Belfast (UK). (Technical Report QUB/IEEE/Tech/ITC2007/CurriculumCTT/v1.0/1)., 2007.

[7] K. Shaker and S. Abdullah, "Incorporating great deluge approach with kempe chain neighbourhood structure for curriculum-based course timetabling problems," in *2nd Conference on Data Mining and Optimization, 2009. DMO '09. 27-28 Oct. 2009*, 2009, pp. 149–153.

[8] K. Shaker, S. Abdullah, A. Alqudsi, and H. Jalab, "Hybridizing Meta-heuristics Approaches for Solving University Course Timetabling Problems," in *Rough Sets and Knowledge Technology*, vol. 8171, P. Lingras, M. Wolski, C. Cornelis, S. Mitra, and P. Wasilewski, Eds. Springer Berlin Heidelberg, 2013, pp. 374–384.

[9] S. Abdullah, H. Turabieh, B. McCollum, and E. K. Burke, "An Investigation of a Genetic Algorithm and Sequential Local Search Approach for Curriculum-based Course Timetabling Problems," *Multidiscip. Int. Conf. Sched. Theory Appl. (MISTA 2009) 10-12 August 2009, Dublin, Irel.*, 2009.

[10] S. Abdullah, H. Turabieh, B. McCollum, and P. McMullan, "A hybrid metaheuristic approach to the university course timetabling problem," *J. Heuristics*, vol. 18, no. 1, pp. 1–23, 2010.

[11] E. K. Burke, B. McCollum, A. Meisels, S. Petrovic, and R. Qu, "A graph-based hyper-heuristic for educational timetabling problems," *Eur. J. Oper. Res. 176 177–192*, 2007.

[12] A. L. Bolaji, A. T. Khader, M. A. Al-Betar, and M. A. Awadallah, "Artificial Bee Colony Algorithm for Curriculum-Based Course Timetabling Problem," *ICIT 2011 5th Int. Conf. Inf. Technol.*, 2011.