Solving University Course Timetable Problem Using Hybrid Particle Swarm Optimization

Ho Sheau Fen @ Irene
Faculty of Comp. Sc. & Info. Sys.
Universiti Teknologi Malysia
81310. Johor, Malaysia
+6019-8331338
ireneluv@hotmail.com

Deris, Safaai
Faculty of Comp. Sc. & Info. Sys.
Universiti Teknologi Malysia
81310. Johor, Malaysia
+6019-7569202
safaai@utm.my

Mohd Hashim, Siti Zaiton
Faculty of Comp. Sc. & Info. Sys.
Universiti Teknologi Malysia
81310. Johor, Malaysia
+6019-7726248
sitizaiton@utm.my

ABSTRACT
Every year or term in a university, each individual department has to design a new timetable for subjects or lessons. The timetabling problems are an optimization problem that consists of allocating a set of lectures in between lecturers and students in a finite period of time (typically a week) that satisfies a set of constraints of various types. Educational timetabling is a major administrative activity at most university and course timetable is one of its types. The course timetabling is known to be highly constrained combinatorial optimization problems. It is a number of courses that are allocated into a number of available classrooms and a number of timeslots, subject to constraints. Many of these problems are tackled in manually which is a tough and time consuming task. The manual solution of the timetable problems usually requires many person-days of work (a two to three weeks for final completion). In addition, the solution obtained may be unsatisfactory in some aspect; for example a student may not be able to take the classes because they are schedules at the same time-slots. The impact of generating a timetable manually was when preconditions change; then whole work becomes unusable (low quality timetable), and has to be restarted from scratch. Therefore, this research is looking into producing a good timetable using a hybrid of particle swarm optimization approach that tackles hard and soft constraints in timetabling problems. The algorithms (hybrid particle swarm optimization-constraint-based reasoning, hybrid particle swarm optimization-local search and standard particle swarm optimization) was validated using real data for university course timetabling problem (UCTP) and finally the conclusion among the compare algorithms will be done.

Categories and Subject Descriptors
G.1.6 [**Constraint Optimization**]: Optimized desire hard and soft constraints while maximizing objective function.

General Terms
Algorithms, Performance, Reliability, Experimentation.

Keywords
University course timetable planning, particle swarm optimization, hybrid particle swarm optimization.

1. INTRODUCTION
The timetabling problem is computationally a NP-complete problem and has been modeled as constraint-satisfaction problem (CSP); therefore, it is very difficult to solve using conventional optimization techniques [6]. Timetabling problem have been widely studied due to their wide range of applications, such as the flight timetable problem, employee timetabling problem, universities or high school timetabling problem. The university course timetabling problem (UCTP) is the planning of allocating a number of events (subjects) into a set of times (timeslots) and resources (rooms) while respecting some other constraints. The basic UCTP consists of finding timeslots for a set of subjects. A set of timetabling tasks is directly related satisfy a set of constraints. In most cases, although solutions generated are acceptable but there are still remain some violation of hard and soft constraints [1], [2].

There are many researcher have introduced different models and algorithms into solving UCTP. Recently, a lot of attention has been paid in automating the construction of timetable planning. The methods that have been tried, including graph coloring [2], [3], simulated annealing [4], tabu search [5], variation of genetic algorithms [6], [7], constraint-based reasoning [1], [11] and particle swarm optimization [8], [9], [10]. Most of the approaches generate feasible but not optimal solutions and particle swarm optimization (PSO) alone hardly controlled the violation of hard and soft constraints. This situation has been noticed during the implementation. To obtain a feasible and near-optimal solution, incorporating PSO and other techniques to handle the constraints has been considered.

In this study, a hybrid approach of PSO and other related constraints handling techniques are tested to solve UCTP. The objective of this study is to compare the hybrid approaches to see which one perform better. To test the algorithms, a real university timetabling data are used and results are presented.

2. UNIVERSITY COURSE TIMETABLING
The UCTP consists in scheduling a set of subjects for a course within a given number of rooms and timeslots. UCTP models must accommodate the characteristics and regulations of specific education systems. Therefore, the problem under consideration varies from universities to universities.

2.1 Structure of the Problem
The data used in this study, is the actual information of the Faculty of Computer Science and Information System, University of Technology Malaysia in Malaysia. The timetable consists of 35 consecutive timeslots and a 9 timeslots per day (five days a week). Each timeslot consists of a 50 minutes, having 10 minutes allocated as break session between subjects, starting at 8:00 AM till 5:00 PM for 5 days. Another 10 extra timeslots are reserved for non-academics activities and lunch hours.

Table 1 illustrates the university course timetable framework used for this paper and Figure 1 shows the structure of weekly university course timetable (UCT). As can be seen from Table 1, the number of subjects are ranging from 1,2,3,…,*n*, *x*-axis represented room 1,2,3,…,*k* while *y*-axis represented timeslots 1,2,3,…,35 [7], [8]. Each rooms and timeslots have its own preferences value depends on the subject's desire features and student's capacity. These are classified as hard and soft constraints.

Table 1. Subject allocation for UCT

Time 35	Subject 1		
.			
.			
Time 3		Subject 2	
Time 2			Subject *n*
Time 1	Subject 3		
	Room 1	**…**	**Room *k***

Time	Monday	Tuesday	Wednesday	Thursday	Friday
8:00 – 8:50	Subject Room	. .	. .	. .	. .
9:00 – 9:50	. .	. .	. .	Subject Room	. .
10:00 – 10:50	. .	Subject Room	. .	. .	. .
11:00 – 11:50	. .	. .	. .	. .	. .
12:00 – 12:50	. .	. .	. .	. .	
13:00 – 13:50					
14:00 – 14:50	. .	. .		. .	
15:00 – 15:50	. .	. .		. .	. .
16:00 – 16:50	. .	. .		. .	. .

Reserved timeslots for non-academic activities and lunch hours

Figure 1. Structure of weekly UCT

2.2 Constraints Definition

The constraints are most important for the UCT to obtain feasible and near-optimal solution. These constraints have to be modeled in a format that can be handled efficiently by the scheduling algorithm. There are two kinds of constraints that are going to be tackled here – the hard constraint and the soft constraint. The constraints considered for the UCTP are:

The *hard* constraints: These are the constraints that must be satisfied no matter what to obtain a feasible timetable. The hard constraints that we are going to focus are as follows:

- Same *subjects* should not be assigned to the same timeslots.
- One *room* should not be assigned to more than one subject for the same timeslot.
- The number of students of a subject assigned to a room should be less than or equal to the *capacity* of the room.

The *soft* constraints: These are constraints of lower priorities to be satisfied. The violation of the soft constraints will not cause the timetable to lose its feasibility. The soft constraints that we are going to focus are as follows:

- The scheduled *time* of the subject should fall within the preference sets as much as possible.
- The scheduled *room* of the subject should fall within the preference sets as much as possible.

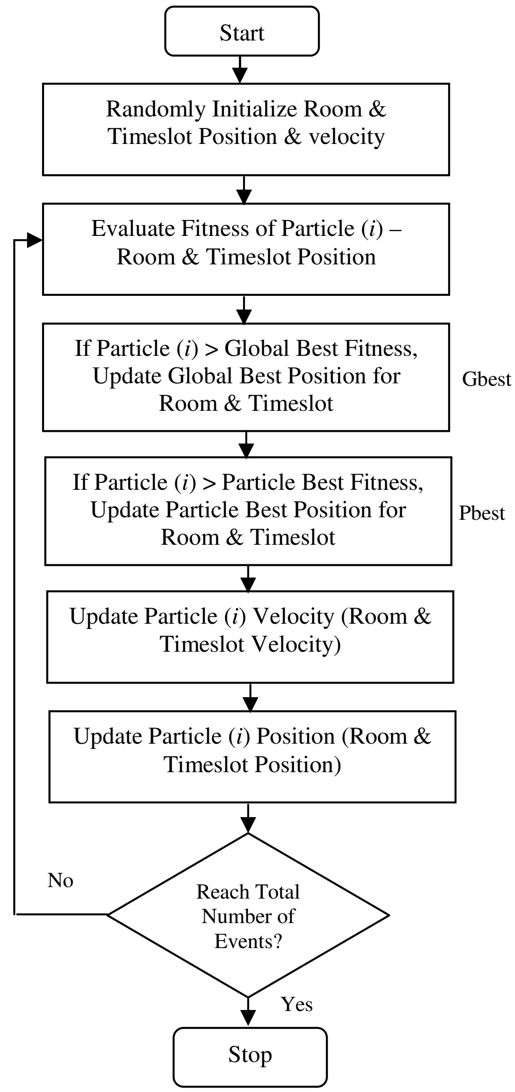

Figure 2. Standard PSO flowchart in UCTP

3. PARTICLE SWARM OPTIMIZATION

The PSO is an evolutionary technique but it differs significantly from genetic algorithms (GAs) [8]-[10]. In PSO, there are no DNA inspired operators on the swarm. Instead, each particle is flying over the search space in order to find promising results and adjusts its flying position according to its' own previous experience and its' neighbor experience.

Assuming that the search space is *D*-dimensional denote by $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$ the *i*th particle of the swarm and by $P_i = (p_{i1}, p_{i2}, \ldots, p_{iD})$ the best position it ever had in the search space. Let *g* be the index of the best particle in the swam and $V_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$ the velocity position of the *i*th particle.

The swarm is manipulated according to the following equations

$$V_{id} = \chi * \left(w * v_{id} + c_1 * r_1 * (p_{id} - x_{id}) + c_2 * r_2 * (p_{gd} - x_{id})\right) \quad (1)$$

$$x_{id} = x_{id} + v_{id} \quad (2)$$

Where d = 1,2,…,D; i = 1,2,…,N and N is the size of the population; w is the inertia weight; c_1 and c_2 are two positive constants; r_1 and r_2 are two random values ranging from [0,1] and χ is a constriction factor used to control magnitude of the velocity.

Equation (1) is used to calculate ith particle's new velocity and equation (2) will update the position of the particle. The performance of each particle is measured using a fitness function (equation (3)). Figure 2 shows the standard PSO flowcharts for UCTP while Figure 3 shows the particle representation for UCTP.

Subject 1	Subject 2	Subject 3		Subject n
Timeslot j Room k	Timeslot j Room k	Timeslot j Room k	…………	Timeslot j Room k

$Subject = 1,2,\ldots,n$, $j = 1,2,\ldots,35$ and $k = 1,2,\ldots,p$

Figure 3. Particle representation for UCTP

3.1 Constraint Satisfaction Problem

CSPs are decision problems defined as set of objects whose state must satisfy a number of constraints [12]. CSP has been classified in two main groups:

- *Complete method* aims at exploring the whole search in order to find all the solutions or to detect none consistent CSP. Concerning to the complete resolution techniques methods, our focus is on backtracking search which is one of the techniques of constraint programming.
- *Incomplete method* mainly relies in the use of heuristics to provide a more efficient exploration of interesting areas of the search space in order to find some solutions. Local search (LS) technique is used in this work.

3.1.1 Constraint-based Reasoning

In artificial intelligence and operational research, constraint satisfaction is the process of finding a solution to a set of constraints. Such constraints express allowed values for variables. A solution is therefore an evaluation of these variables that satisfies all constraints. [13] Usually a number of rules are defined for assigning resources to events. When the propagation of the assigned values leads to an infeasible solution, a backtracking process enables the reassignment of value(s) until a solution that satisfies all of the constraints are found [14].

3.1.2 Local Search Algorithm

Local search is an iterative algorithm that moves from candidate solution S to neighbor solution S' according to some neighborhood structure [15]. Local search procedure usually consists of the following steps.

Step 1, choose an initial solution S to be the current solution.

Step 2, select a neighbor S' of the current solution S.

Step 3, test whether to accept the solution from S to S'. If the solution is accepted, then S' replaces S as the current solution; otherwise S is retained as the current solution.

Step 4, test whether the algorithm should terminate. If it terminates, output the best solution generated; otherwise, return to step 2.

3.2 Hybrid Particle Swarm Optimization

Over at this study, two different kinds of hybrid PSO are used for experiment purpose. The first one is hybrid PSO-local search (LS) and the second one is hybrid PSO-constraint-based reasoning (CBR).

3.2.1 Hybrid PSO-LS

This hybrid algorithm is running under 2-phases, the PSO and local search. At each iteration, a subject would be scheduled into the timetable by using the PSO strategies where particles are looking for the best position (room and timeslot) and when clashes occurred local search will be apply to seek for nearest available neighborhood with highest timeslot and room preference. Best positions mean allocating a subject into preferred room and timeslot without violating hard constraints and minimize the violation of soft constraints. The process will repeat until all subjects are allocated. Figure 4 shows the flowchart of hybrid PSO-LS in UCTP.

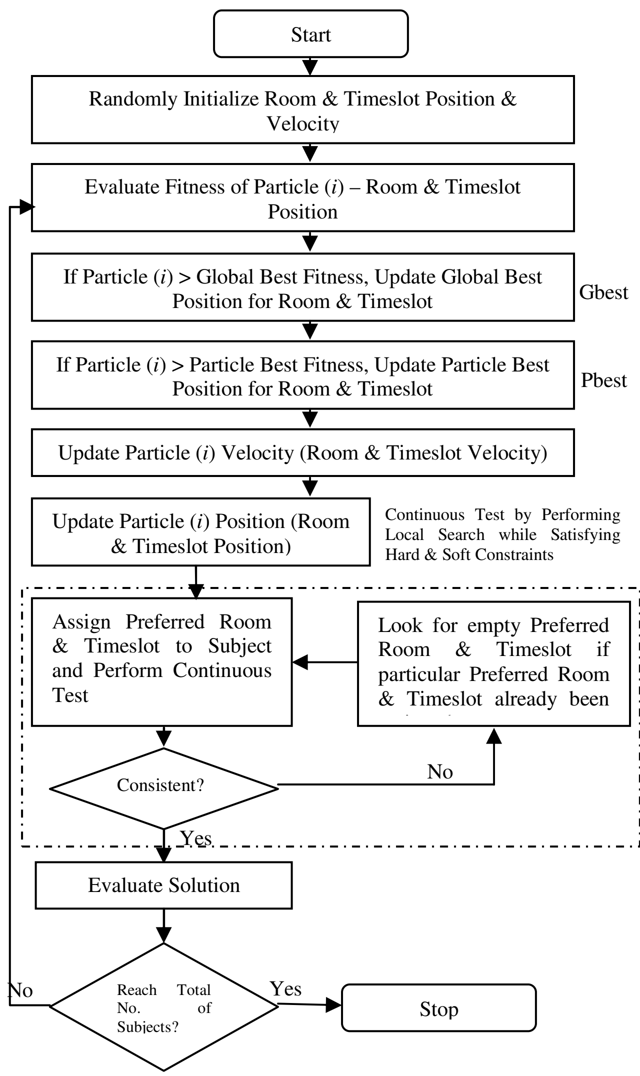

Figure 4. Flowchart of hybrid PSO-LS in UCTP

3.2.2 *Hybrid PSO-CBR*

PSO is known to have the ability to find a near-optimal solution by updating its flying position and velocity vector with a suitable fitness function. [17]-[19] has reported that a PSO alone is inadequate to solve complex problems such as optimization of the constraint satisfaction problems (CSPs). This is because of the nature of the PSO itself, where it has been designed to find potential solutions by updating its' position and velocity based on a specific fitness function. Therefore, a technique that process constraints is needed to find a near-optimal solution for the UCTP. The CBR which has been designed specially to handle constraint satisfaction through its arc-consistency algorithm is found [6]. Thus, by combining PSO and CBR, it can efficiently optimize the UCT constraints. This hybrid method is proposed by incorporating the CBR procedure to process and validate each particle position (room and timeslot) generated by the PSO. Through this method, more accurate particle position is taken as final solution. Figure 5 shows the flowchart of hybrid PSO-CBR in UCTP. This representation enables both consistent checking and search for a near-optimal solution incorporating CBR and PSO techniques. Detail description of these improvements is discussed in the following section.

3.2.2.1 *Constraints-Processing Algorithms*

The main objective of constraint-processing procedures is to test whether the particle position (room and timeslot) supplied by PSO satisfies the desire constraints. If the room and timeslot value is valid, then it will be selected for that particular event. Otherwise, a backtracking procedure will be performed to select next room and timeslot based on preferred score [11], [16]. This procedure can be regarded as a repair procedure to ensure feasibility of selected combinations obtained by PSO. Figure 6 shows the example of backtracking search tree.

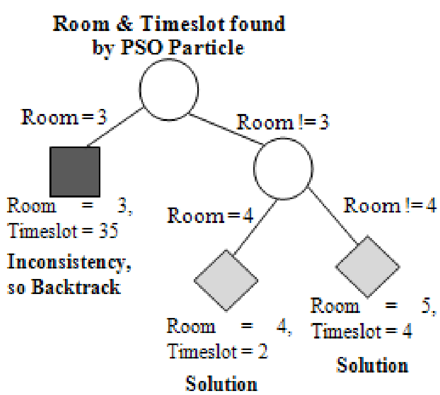

Figure 6. Example of backtracking search tree

3.3 Fitness Function

The fitness function is needed to optimize the preferences of utilizing a good timeslots and rooms [16]. This is because room has different kind of facilities, accessibility and locations. As for timeslots, all good timeslots should be selected first before the less favorable timeslots are chosen. Thus, the fitness function *f(t)* of a timetable *t* is given as follow:

$$\boldsymbol{f(t)} = \sum_{i=1}^{n}\left(\boldsymbol{P}(\boldsymbol{T}(\boldsymbol{S_i})) + \boldsymbol{P}(\boldsymbol{R}(\boldsymbol{S_i}))\right) \quad \textbf{(3)}$$

Where $P(T(S_i))$ are timeslot preferences for subjects S_i, $i=1,2,\ldots,n$ and $P(R(S_i))$ are room preferences for subjects S_i, $i=1,2,\ldots,n$. The fitness function *f(t)* for each particle (solution) is computed by the total sum of preference value of timeslot $P(T(S_i))$ and preference value of room $P(R(S_i))$ for subject S_i, $i=1,2,\ldots,n$. This function enables a search for a near-optimal and feasible solution with the highest preferences value of *f(t)* and is designed to maximize the of rooms and timeslots preference values.

4. EXPERIMENTAL AND RESULTS

The algorithms have been tested using timetable data from Faculty of Computer Science and Information System, University of Technology Malaysia. Table 2 summarizes the information for this university timetable. The parameter settings for the PSO reported at Table 3 while Table 4 and 5 summarizes the timeslot and room ordering and preferences. The implementation has been developed in C. Experimental results are obtained by using notebook which has Intel Core 2 Duo 2.2 GHz processor and 2 GB RAM.

Table 2. Summary of the university timetable

Resources	Value
No. of Subjects	183
No. of Rooms	21
Total Timeslots	45
Total Timeslots Reserved	10
Total Timeslots Available	35

Table 3. Parameter of PSO

Parameters	Value
No. of Generations	400
No. of Particles	10
c_1	2.8
c_2	1.3
w	1 / (2 * log (2))

Table 4. Timeslots ordering and preferences

Timeslot no.	No. of lectures per week	Preference score	Order
1,2,3,4	4	3	1,2,3,4
5,6,7,8	4	2	5,6,7,8
9,10,11	3	3	9,10,11
12,13,14	3	3	12,13,14
15,16,17	3	3	15,16,17
18,19,20	3	2	18,19,20
21,22,23	3	1	21,22,23
24,25,26	3	1	24,25,26
27,28,29	3	1	27,28,29
30,31	2	3	30,31
32,33	2	3	32,33
34,35	2	2	34,35

Table 5. Rooms ordering and preferences

Room no.	Capacity (student no.)	Preference score	Order
1	20	2	1
2	20	1	2
3	30	2	3
4	30	2	4
5	30	2	5
6	30	1	6
7	30	1	7
8	50	1	8
9	60	2	9
10	60	2	10
11	60	2	11
12	60	2	12
13	60	2	13
14	60	1	14
15	60	1	15
16	60	1	16
17	60	1	17
18	60	1	18
19	150	2	19
20	150	1	20
21	150	1	21

The results for standard PSO algorithm is shown in Table 6 while Table 7 and 8 show the results for hybrid PSO-LS and hybrid PSO-CBR algorithm. The three hybrid algorithms have been tested under 5 separated runs.

Table 6. Results after 5 runs for Standard PSO

Seed	No. of subjects unallocated	Time (Sec)	Max. value of objective function
1	89	17.00	353
2	86	17.00	382
3	78	17.00	388
4	86	17.00	380
5	81	17.00	384
Average	**84**	**17.00**	**377.4**

Table 7. Results after 5 runs for hybrid PSO-LS

Seed	No. of subjects unallocated	Time (Sec)	Max. value of objective function
1	0	16.00	720
2	0	17.00	724
3	0	17.00	720
4	0	18.00	721
5	0	17.00	715
Average	**0**	**17.00**	**720**

Table 8. Results after 5 runs for hybrid PSO-CBR

Seed	No. of subjects unallocated	Time (Sec)	Max. value of objective function
1	0	17.00	743
2	0	17.00	742
3	0	18.00	740
4	0	17.00	740
5	0	17.00	744
Average	**0**	**17.20**	**741.8**

From Table 8, by hybrid PSO-CBR the average value of objective function generated is greater than those obtained by standard PSO and hybrid PSO-LS, shown in Table 6 and Table 7, respectively. This is because the hybrid PSO-CBR backtrack the search if invalid solution is found. The average computational time used to generate a solution for hybrid PSO-CBR also slightly longer compared to hybrid PSO-LS and standard PSO due to its backtracking procedure that backtracks to search for consistent solution when inconsistent solution found. Besides that, the standard PSO have a total average number of 84 subject's unallocated compare to both hybrid PSO-CBR and PSO-LS that successfully allocated all the subjects. The main reason that the standard PSO still has subjects unallocated because it is doesn't have the capability to handle constraints like both hybrid PSO-CBR and hybrid PSO-LS does. Figure 7 and Figure 8 show the utilization of rooms and timeslots from hybrid PSO-CBR, hybrid PSO-LS and standard PSO algorithms.

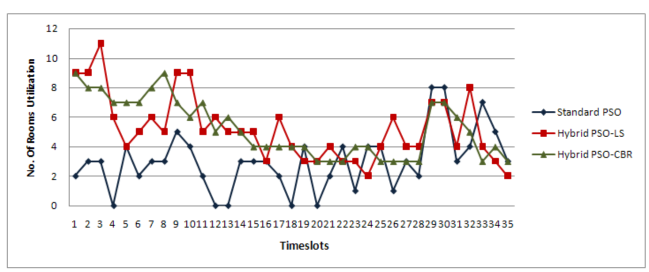

Figure 7. Rooms utilization for the three algorithms

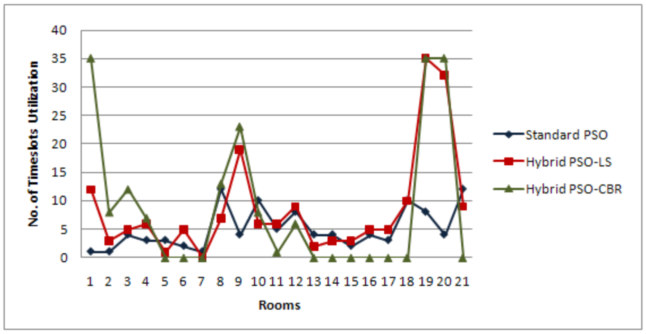

Figure 8. Timeslots utilization for the three algorithms

Based on Figure 7, it can be seen that, by using hybrid PSO-CBR all early timeslots are chosen for the subject allocation first before the less preferred one is chosen. Because hybrid PSO-LS does not have backtracking ability, the allocation of event is allocated as long as hard constraints are not violated. Thus, it can be seen that the graph looks a little inconsistent. Some of the rooms are not utilized by using standard PSO algorithm because it has no ability to handle constraints so when there is clashes occur, the algorithm just count the unallocated subjects and continue allocating the rest of the subjects.

Figure 8 shows that the hybrid PSO-CBR algorithm attempted to utilize all the preferred timeslots and rooms and it can be seen that there are several rooms left unused. It is a good sign where it can be kept for future usage due to the increments of students and subjects annually. Based on the fitness function chosen, the hybrid PSO-CBR algorithm can produce a feasible and near-optimal solution. Hybrid PSO-LS produced feasible solutions but it utilized all the resources and because of its' doesn't have the backtracking search, its' accept all possible solution. Thus, the lowest room preferences value also been chosen for allocation. Same goes to standard PSO but this algorithm has worsened the case whereby it still has some unallocated subjects.

5. CONCLUSION AND FUTURE WORK

This paper studied on the hybrid PSO-CBR, hybrid PSO-LS and standard PSO approaches in solving UCTP. From the experiments, it indicates that all techniques provide feasible solution but to have a near-optimal one with acceptable computational time, hybrid PSO-CBR is more promising. With the use of hybrid PSO-CBR approach in UCTP, the rooms and timeslots preferences are maximized with the help of fitness function used. The CBR in the hybrid algorithm maintained valid values of the PSO flying position through constraint posting and backtrack the search if there's any invalidity. This study also shows that the convergence of optimizing UCTP is efficient due to CBR ability to significantly reduce the search space. Hence, the hybrid PSO-CBR algorithm is capable in finding feasible and near-optimal solution compare to hybrid PSO-LS and standard PSO algorithm. Future research work will be focusing into experimenting different scheduling area using the hybrid PSO-CBR algorithm and enhancement of the algorithm will be made base on the problem that need to be solve.

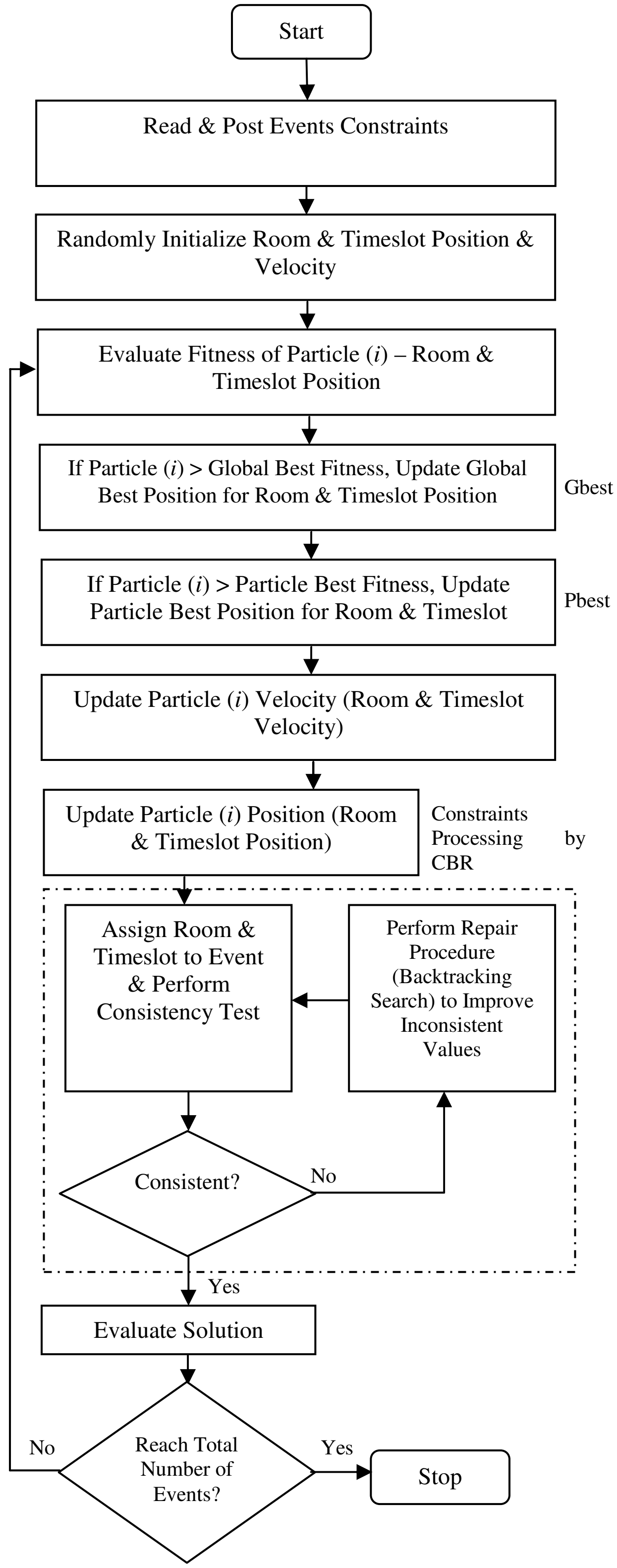

Figure 5. Flowchart of hybrid PSO-CBR in UCTP

6. ACKNOWLEDGMENTS

Malaysian Ministry of Science, Technology, and Innovation (MOSTI) and Universiti Teknologi Malaysia Research Management Centre (RMC) for their support (research grant number: Vote 78257).

7. REFERENCES

[1] Tuncay, Y.: Constraint-based School Timetabling Using Hybrid Genetic Algorithms. AI*IA 2007: Artificial Intelligence and Human-Oriented Computing. LNCS, vol. 4733, pp. 848--855. Springer, Heidelberg (2007)

[2] de Werra, D.: An introduction to timetabling. European Journal of Operations Research 19, pp. 151--162 (1985)

[3] Burke, E. K., D.G. Elliman., R. Weare.: Automated Scheduling of University Exams. Proceedings of the IEEE Colloquium on Resource Scheduling for Large Scale Planning System. (1993)

[4] Adora, E. C., Augusta, Y. H., Bobby O. C. JR.: Parallel Hybrid Adventures with Simulated Annealing and Genetic Algorithms. Proceedings of the International Symposium on Parallel Architectures, Algorithms and Networks. (2002)

[5] Chu, S. C., Fang, H. L.: Genetic Algorithms vs. Tabu Search in Timetable Scheduling. Knowledge-Based Intelligent Information Engineering Systems, Third International Conference, pp. 492-495. Adelaide, SA, Australia (1999)

[6] Safaai, D., Sigeru, O., Hiroshi, O., Puteh, S.: Incorporating Constraint Propagation in Genetic Algorithm for University Timetable Planning. Engineering Applications of Artificial Intelligence 12, pp. 231-253. (1999)

[7] Sigl, B., Golub, M., Mornar, V.: Solving Timetable Scheduling Problem Using Genetic Algorithms. The 25th International Conference Information Technology Interfaces ITI 2003. Cavtat, Croatia. (2003)

[8] Chu, S. C., Chen, Y. T., Ho, J. H.: Timetable Scheduling Using Particle Swarm Optimization. Proceedings of the First International Conference on Innovative Computing, Information and Control. (2006)

[9] Tan, W. N.: A New Examination Timetabling Algorithm. 16th Proceeding of National Symposium Mathematics and Science. (2008)

[10] Fealko, R. D.: Evaluating Particle Swarm Intelligence Techniques for Solving University Examination Timetabling Problems. Graduate School of computer and Information Sciences, Nova Southeastern University: A Dissertation for the degree of Doctor of Philosophy. (2005)

[11] Legierski, W.: Constraint-based reasoning for timetabling. Artificial Intelligence Method: AI-METH, Gliwice, Poland. (2002).

[12] L. Tony, M. Eric, and S. Frederic, "Hybridization Strategies for Local Search and Constraint Propagation," Proceeding of the 4th Workshop on Cooperative Solvers in Constraint Programming, September 2004.

[13] Muller, Tomas., Rudova, Hana., and Bartak, Roman., "Minimal perturbation problem in course timetabling," In Edmund Burke and Michael Trick, editors, Practice and Theory of Automated Timetabling, Selected Revised Papers, Springer-Verlag LNCS, 2005.

[14] Petrovic, S. and Bykov, Y., "A Multiobjective Optimisation Technique for Exam Timetabling Based on Trajectories," in E.Burke, P. De Causmaecker (Eds.) Practice and Theory of Automated Timetabling IV, Lecture Notes in Computer Science 2740, Springer 2003, pp. 179-192.

[15] A. Schaerf, "Local search techniques for large high-school timetabling problems", in IEEE Transactions on Systems, Man, and Cybernetics 29, volume 4, 1999, pp. 368-377.

[16] S. Deris, S. Omatu, and H. Ohta, "Timetable Planning using the Constraint-based Reasoning," *Computer & Operations Research*, 27, 2000, pp. 819-840.

[17] Irene, Sheau Fen Ho., Deris, Safaai. and Mohd Hashim, Siti Zaiton., "A Study on PSO-based University Course Timetabling Problem" in IEEE 2009 International Conference on Advanced Computer Control (ICACC 2009), pp. 648-651, Singapore.

[18] Irene, Sheau Fen Ho., Deris, Safaai. and Mohd Hashim, Siti Zaiton., "A Combination of PSO and Local Search in University Course Timetabling Problem" in IEEE 2009 International Conference on Computer Engineering and Technology (ICCET 2009), pp. 492-495, Singapore.

[19] Irene, Sheau Fen Ho., Deris, Safaai. and Mohd Hashim, Siti Zaiton., "Tackling University Course Timetable Constraints using the Combination of Swarm Intelligence and Local Search" in IEEE Proceedings of 2008 Student Conference on Research and Development (SCOReD 2008), pp. 205-1 – 205-5, 26-27 November 2008, Johor, Malaysia.