Journal of Theoretical and Applied Information Technology
20th February 2015. Vol.72 No.2
© 2005 - 2015 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

# MULTIPLE-HYBRID CASE-BASED REASONING APPROACH FOR UNIVERSITY COURSE TIMETABLING PROBLEM

[1]HONG SIAW THENG, [2]ABU BAKAR BIN MD SULTAN, [3]NORHAYATI MOHD ALI
Faculty of Computer Science and Information Technology, University Putra Malaysia, 43400 UPM, Serdang, Selangor, Malaysia
E-mail: [1]archer.hong.82@gmail.com, [2]abakar@fsktm.upm.edu.my, [3]norhayati7@gmail.com

**ABSTRACT**

The University Time Tabling problem (UCTP) is an allocation process to schedule courses with available time slots for suitable lecture halls, classes or rooms in every semester. The academic time tables design has become a very complex issue and difficult task as such it is classified as NP-hard problem. The objectives of this paper is to achieve higher accuracy for time tabling plotting as well as improve the effectiveness while generating academic courses time tables. With the aim to improve timetabling solution, we investigate into a new algorithm which is based on Case-based Reasoning (CBR) retrieval function. The idea proposed is known as Human Preference Adaptable Retrieval Approach – HPARA and is separated in three parts, with the combination of different functionality: Prioritized Attributes, Frequency Grouping, and Value Difference Measurement. The CBR retrieval is a strategy to retrieve similar cases from previous time tabling for generating new time tables. Retrieved cases will then filtered with the according functionality: Prioritized Attributes, Frequency Grouping, and Value Difference Measurement too seek for the most suitable components. The proposed CBR algorithm is tested with the database of Faculty of Computer Science and Information Technology for the past five years compared with Genetic Algorithm (GA). Experimental results show promising improvement for both accuracy and effectiveness with the proposed CBR algorithm to generate UCTP.

**Keywords:** *Timetabling, Case-based reasoning, Case-based retrieval, Multiple-hybrid, HPARA*

## 1. INTRODUCTION

University timetabling is a major administrative activity for a wide variety of institutions. Thus, the activity of timetabling is a problem of assigning a number of events into a limited number of time periods. A real timetabling can exist in various forms or patterns like the most common educational timetabling; which may divide into two main categories: course timetabling and exam timetabling, follow with the sports timetabling, events timetabling, transportation timetabling and shift rotation timetabling etc. In the process to generate a timetabling to satisfy the requirements for certain condition, many difficulties may occurred with the variance of constraint and diversity of different occasions. Therefore, in order to generate a constraint satisfaction timetabling, automated timetabling is a very important task compared to man-made timetabling. Automated timetabling can greatly reduce time for resource arranging, searching and scheduling. This saves a lot of man-hours working not only to institutions but to other organizations as well as companies. The benefits include increase the quality of educations in institutions if arrangement is well met as such constraint is optimized. Usually, constraints are divided into two different group which known as hard constraint and soft constraint. Hard constraints are the requirements that must be fulfilled or satisfied. Only if all hard constraint requirements are met, a time table will be consider as a feasible one. As an example for the common hard constraint: "No lecturer will be teaching two different courses at the same time". Soft constraints are those requirements which are not essential for the timetabling feasibility. However, it is highly desirable for timetabling to meet soft constraint to produce more effective results. Many of the algorithmic technique

© 2005 - 2015 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

for timetabling were derived directly from graph colouring-based heuristics; Rhydian Lewis, 2007 [1]. However, with evolution of various techniques to improve the optimization, much interest had been done in metaheuristics. With the nature of metaheuristics : "..can be seen as a general algorithmic framework which can be applied to different optimization problems with relatively few modifications [being needed] to make them adapted to a specific problem" [1], it is widely applied to timetabling solution with the characteristics of idiosyncratic nature. The five main metaheuristics paradigm: Evolutionary Algorithms, Simulated Annealing, Tabu Search, Local Search and Ant Colony Optimizations.

Case-based reasoning has been applied to scheduling problems since the year 1997 by Pádraig Cunningham et al [2]. Since then, more research had been done focus on using CBR with different techniques for timetabling optimization.

In this paper, we present a new CBR retrieval function by introducing the combination of three different approaches in order to increase the accuracy and effectiveness in solving timetabling problems. The paper is organized as follows. Section 2 explains the background and motivation for this research. Section 3 explains the methodology of the new CBR retrieval approach known as the Human Preference Adaptable Retrieval Approach – HPARA. Section 4 shows the experimental design and process and finally section 5 provides results and discussion. This work is important to justify the enhancement of human preference case retrieval in CBR system which aim to improve retrieval accuracy and efficiency in CBR problem solving.

## 2. BACKGROUND AND MOTIVATION

Several researchers have explored different approaches to solve the timetabling problems. For instance, Beligiannis et al. [3][4] apply the genetic/evolutionary algorithms (EA) with the adaptive behavior in EA to specified constraint and assigning weight to satisfy the time table results. Ross et al. [5] discussed the state of the arts of variety of representations and algorithms in solving timetabling problems. Nediah et al. [6] improves pattern matching using genetic programming to evolve the most adequate traversal order which improve space usage and matching times. Qarouni-Fard et al. [7] using particle swarm optimization to classic timetabling problem based on social-psychological principles. Burke et al. [8] developed tabu-search hyperheuristic for timetabling optimization; with heuristics compete using rules based on the principles of reinforcement learning. Di Gaspero et al. [9] present algorithms based on tabu-search which import several features from the research on the Graph Colouring problem. Socha et al. [10] proposed Ant-System for simplification of university course timetabling known as MAX-MIN Ant System, which make use of a separate local search routine. Finally, Burke et al. [11] employed structured cases with CBR retrieval focus on searches for structurally similar cases in case base.

From the observation of academic timetabling in every semester, the changes are not always a major transformation. Instead, most of the time, time table plotting only required some minor alteration or transition of certain components within timetables. Thus, as a practice for constructing a timetable, it is normally start from ‘‘last year’s’’ timetable and make changes as less as possible. This is only appropriate if there is little change in the problem from one year to another. From the behavior of revise previous cases to generate new case, it provided the motivation for investigating a CBR approach to timetabling problems.

### 2.1 Case-based Reasoning

CBR is a paradigm, concept and instinctive for problem solving. It is also a process to solve new existing problems by referring back and adapting previous case solutions to fit the new situations. “Past cases will be revised before applying to the new cases. In any case of the successful problem solving cases will be retaining for future revise and reuse” [12]. However, CBR also could have a different meaning depend on the intended usage of the reasoning. Koldner claim that CBR is an Artificial Intelligence methodology that is based on using previous problem solving experience in solving new problems [13]. A CBR could mean to adapt and combine old solutions to solve a new problem, explained new situations according to experienced cases or situations, critique new solution base on old cases, reason from precedents to understand new situations, or build a consensuses solution based on previous cases.

The well known classical and traditional CBR system is known as the R4 model, which constitute the following four processes: *retrieve,*

Journal of Theoretical and Applied Information Technology
20th February 2015. Vol.72 No.2
© 2005 - 2015 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

*reuse, revise, retain*. Below is the review of a few models of CBR process cycle.

### 2.1.1 R4 Model of CBR

Introduced by Aamodt and Plaza, which is the general CBR Model known as the R4 Model. The reason to be known as R4 is because of the processes involved in this model is represented by 4 Rs. Figure 1 represent the CBR R4 Model.

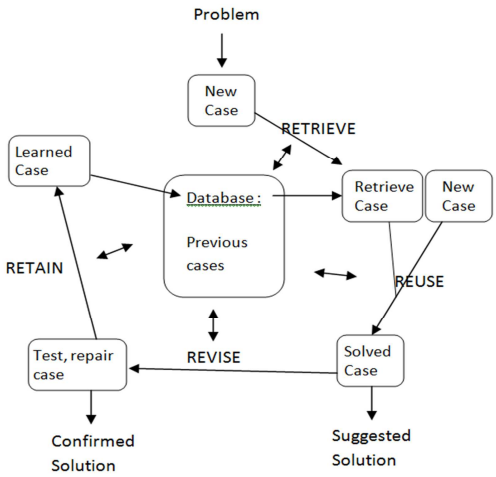


*Figure 1: CBR R4 Model*

### 2.1.2 Hunt's Model of CBR

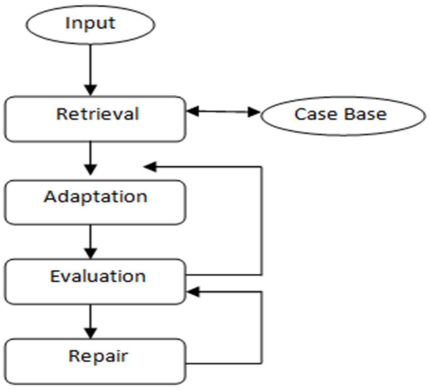


*Figure 2: Hunt's Model*

From Figure 2 by J. Hunt's Model of CBR [14], the first step performed by the CBR system is to analyze inputs to the system once a case has been obtained and determine the important feature to use by selecting from the past cases in the case base. Features are then passed to the retrieval step along with the initial inputs from case obtained. Retrieval step will then use the features to match cases in case base to select cases into a list to match the current situation. Once the best match has been retrieved to match the current problem, the adapted case will be evaluated as if it provides solution for the current problem. At evaluation step, if the solution is accepted, then it'll be presented as the solution and stored in the case base for future use. If some aspect of the case is not yet solved by the solution, then it must be repaired such that all aspect of the problems is addressed.

### 2.1.3 Kolodner and Leake's CBR process Model

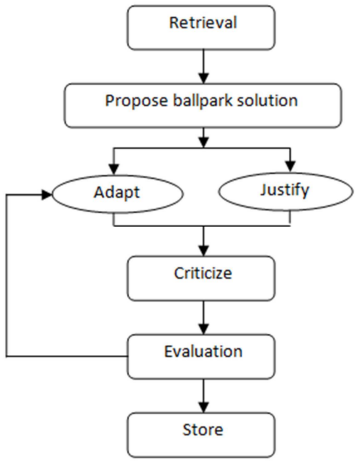


*Figure 3:. Kolodner and Leake's Model*

In Kolodner and Leake's theory (from Figure 3), they consider CBR as a process of 'remember and adapt' or 'remember and compare' in their propose CBR cycle. From the process model, case retrieval is the primary process. However, partially matched case must be retrieved to facilitate reasoning. Retrieval process depends on choosing appropriate indexes and to search for the relevant cases in the case base. Reasoner will criticize candidate solutions to identify potential problems in order to make sure poor solutions will not repeated with the good solutions on the new cases. The reasoner also must be able to evaluate its performance base on external feedback in order to become more proficient. After feedback is analyze, cases will be updated and the outcomes will be recorded. Cases that were used to solve the problem will be reindexed based on analysis of their usefulness.

Journal of Theoretical and Applied Information Technology
20th February 2015. Vol.72 No.2
© 2005 - 2015 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

### 2.2 Case-based Reasoning in Timetabling

CBR has been successfully applied to various scheduling problems including the planning and scheduling of large-scale airlift operations [15], dynamic job-shop scheduling [16], repair problems in job-shop scheduling [17] and production planning and control problems [18]. These applications indicate that CBR is potentially a valuable tool in solving timetabling problems.

Following the basic idea behind case-based reasoning [13], a number of previously solved timetabling problems are stored in a case base. These case bases are later used for constructing solutions for new timetabling problems. Since then, a number of researches had been done which applied CBR in educational timetabling problems. Burke et al. [19], apply structured cases in CBR which reusing and adapting cases for timetabling problems using attribute graph approach. The research is the further developed in the year 2001. Burke aimed to solve a wider range of problems with similarity measures [20]. At the year of 2005, the author enhanced the approach again which is known as multiple retrieval CBR that partitions a large problem into small solvable sub-problems by recursively inputting the unsolved part of the graph into the decision tree for retrieval [21]. In the year 2006, Burke release another approach known as Case-based heuristic selection for timetabling problems which aimed to increase the generality [22].

From the review of previous researches, the importance of timetabling solving using CBR lies within the concept of case retrieval. The improvement of CBR retrieval approach was always aimed to increase the generality. The above observation induces the idea of combining the important features such as similarity measurement, prioritized attributes and frequency grouping. To increase an approach's generality, human preference is the key problems to determine whether a system is helpful base on different specifications and requirements. A system is claimed to be high generality while requirements from different aspect able to handle and produce a result as accurate or matching one's requirements. Thus, retrieval approach which able to adapt with human preference is important to produce a feasible time table.

## 3. RESEARCH METHODOLOGY

The methodology used for this paper is an actual test bed. Thus, gathering previous time tables from actual cases is the most priority. In this paper, the actual cases are gathered from past ten semesters of time tables from Faculty of Computer Science and Information Technology, University Putra Malaysia which categorized into four departments which are Multimedia, Computer Science, Information System, and Computer Networking respectively. Then, the gathered time tables are restructured into Database Management System (DBMS) format with appropriate links. Figure 4 shows the Entity Relational Diagram (ERD).

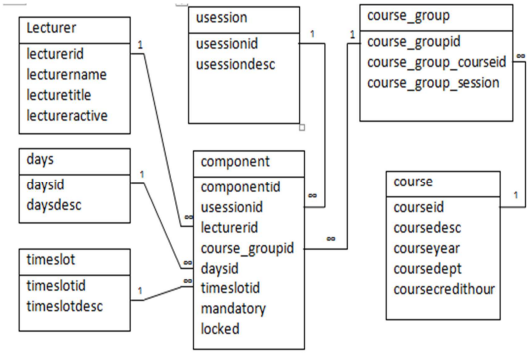


*Figure 4: Database relation table*

The most important table is the "*Component*" table which will be keeping all the scheduling objects. *Component* table is an associative table which has one-to-many relation with all other tables. The 'mandatory' field in *Component* is to determine user input (hard-constraint) object, if 'mandatory' field is TRUE, then this object is not allow rescheduling with algorithm. Whereas if 'mandatory' field is FALSE, it indicates this object is one of the results of scheduling algorithm. The 'locked' field in *Component* is to prevent any alteration of the object, mainly for previous cases. If 'locked' field is flagged TRUE, this object will be allowed to refer as previous case. The 'lectureractive' field in *Lecturer* is to identify if the lecturer is currently available for this session scheduling, this field can be changed for every different session. This field is temporary set for scheduling algorithm purpose only. It doesn't reflect the lecturer is active/not active throughout every session. Scheduling doesn't directly schedule with course, instead schedule with *Course_group.* Each course can have more than one group, and each group has to be scheduled with different time resources. *Course* is fixed for all

© 2005 - 2015 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

sessions while *Course_group* has to specify in every session. Scheduling algorithm will schedule based on *Course_group*. If all objects of a selected session in *Course_group* are scheduled, then the scheduling is considered complete and will stop.

This paper experiment is built with a PHP web-based scheduler. The scheduler allows multiple algorithms plug into it and compare results of scheduling. Web-based has been chosen is because it's cross-platform and ease of use. This experiment tool (scheduler) can later modify into an actual web-service as a time tabling system. The DBMS of this experiment is MySQL, and the webhosting service is APACHE. The conversion of the hardcoded text into database is by using Regular Expression Replacement (RegEx). And the Integrated Development Environment (IDE) is Notepad++. The whole experiment is based on actual previous databases and will schedule result to compare for current accuracy, thus this experiment is a real-time test bed. The experiment will be performed on a high-performance Desktop. The performance specification of the desktop is 3.4 Ghz i7-Quad Core (Hyper-thread) Intel processor and 8GB RAM.

Experiments were conducted base on the schedules database of Faculty of Computer Science and Information Technology, University Putra Malaysia from the year 2008 until year 2012 with a total of 9 semesters of schedule from 4 different departments.

The main parameters used in this scheduling experiment are Time, Day, Lecturer, and Course group. These four parameters will join up as a component. While each component will labeled with a session to indicate such combination has actually used in the specified session. Reusing the component will only refer these four parameters by labeling a new session. Retrieving all components with the same session will form up as a Previous Case. Hard constraint and Soft constraint will be specified during the experiment.

The existing algorithms will be taken into the experiment for comparison would be Genetic Algorithm. The proposed algorithm for this experiment that will be plug into this scheduler is Case-Based Reasoning Algorithm with HPARA approach. All the above algorithms will perform the same task providing the same existing database. The performance metrics will be used to evaluate each algorithm would be Accuracy of result and Time taken to schedule the result. The results of all algorithms will be collected and plot into graphs. Conclusion will be drawn through the graphs.

## 4. HUMAN PREFERENCE ADAPTABLE RETRIEVAL APPROACH – HPARA

There're many other approaches with different types of mathematical inspiration to implement in similarity measurement. However, none of the researches have considered about prioritize attribute similarity, which is why CBR so far unable to widely implement in those precision-sensitive application. In most CBR retrieval and similarity measurement research, researchers consider the attributes are equally important in case matching. However, in actual application, depending on users' desire, attributes' priorities are always bias due to human imbalance factor. Hence, allowing users to prioritize their own attributes is believed to greatly help in improving CBR retrieval accuracy. The first idea proposed in this paper is by considering attributes' priorities.

Most CBR retrieval approaches greatly depends on user input which the desired target case is. The flow of CBR would be creating a desired target case, then retrieve old cases by measuring similarity, and then matched case would be taken and refine if necessary to be the target case result. However, there're some users that not even know their own desire target case. In this case, it'd be difficult for CBR to perform. In this paper, the second proposed idea would be searching the old cases by grouping the identical and/or similar cases and find the most frequent match to become the target case. The search can be from one to any number of attributes depending on users. The result will always be the most frequently used group that only varies depending on number of attributes used. The third idea of this paper is to combine all three approaches which are prioritized attributes, frequency grouping, and conventional similarity measure into a new algorithm. This algorithm is expected to extract a very accurate result provided the database is large enough. This algorithm is also very portable as it's customizable for all the three approaches. Figure 5 shows the HPARA retrieval cycle.

Journal of Theoretical and Applied Information Technology
20th February 2015. Vol.72 No.2
© 2005 - 2015 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

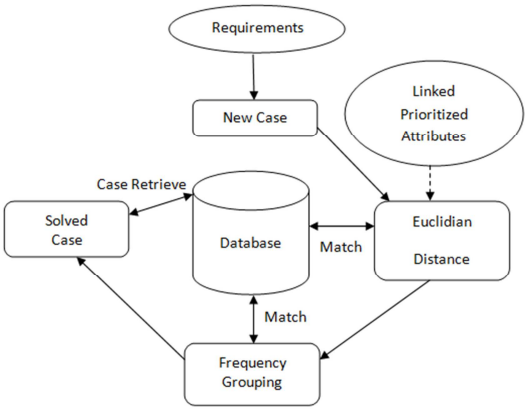


*Figure 5:. HPARA Retrieval cycle*

### 4.1 Linked Prioritized Attributes

This approach seeks not only one attributes but it prioritized a few main attributes in the cases to match with requirements. In order to make certain in better accuracy for finding most appropriate cases, bonus point is given for the case which hit two or more attributes in the same time. Consider following Table 1 below as cases with 5 attributes.

*Table. 1: Example cases for Linked Prioritized Attributes*

| | Subject | Lecturer | Day | Time | Classroom |
|---|---|---|---|---|---|
| **Target Case** | Programming | Ali | Mon | 0800 | 101 |
| **Case A** | Programming | Ali | Tue | 1700 | 203 |
| **Case B** | Programming | Michael | Mon | 0800 | 101 |
| **Case C** | Programming | Ali | Mon | 0900 | 304 |

Let's use a simplest similarity measurement on the above case study which gives 1 point for each matched attribute. Table 2 shows the similarity points for all the cases:

*Table 2: Case match points*

| | |
|---|---|
| Case A | 2 points |
| Case B | 4 points |
| Case C | 3 points |

Thus, in this case, Case B is considered the most matched case and will be retrieved to use as result. Case B is definitely the most matched case if user's bias does not take into account. However, let's put it this way, Michael has taught Programming once thus it has the record in the database, but Michael shows bad performance in that class and the faculty decided not to let him teach Programming anymore while Ali shows significant performance in lecturing Programming subject. At this point, users will see Case B is a very inaccurate result while users might not know how the algorithm works behind hence user concluded CBR unable to give the accurate result. Users would prefer Ali lectures Programming subject regardless the Day, Time and/or Classroom Ali is going to give the lecture. So this will result the users expecting to see the retrieved case either is Case A or Case C.

However, if priority point is considered into the similarity measurement to the above case study, the outcome may be different comparing to previous case pattern. The condition of this point would be if Subject match at the same time with Lecturer, then an extra 3 points will be given. The priority points are allowed to adjust by users. Table 3 shows the similarity points with prioritized attributes for all the cases:

*Table 3: Case match points with prioritized attributes*

| | |
|---|---|
| Case A | 5 points |
| Case B | 4 points |
| Case C | 6 points |

At this point, the returned result would be Case C which has 6 points of similarity even though it has only matched 3 attributes compared to Case B which has matched 4 attributes. Let's consider the case, if Case C is not exists in the database, then the result would still be Case A instead of Case B even though Case A has only 2 matched attributes. Comparing Case A and Case C, certainly if both cases have matched the prioritized attributes, then the more matching would be considered more similar. This approach allows the user to set more than one priority rules which can greatly contribute to case retrieval accuracy.

### 4.2 Frequency Grouping

Frequency Grouping approach is selecting a list of attributes to form a group for matching the most frequently used in the database. The SQL for this search is as follow:

```
SELECT count(*) as `Frequency`, [attr1], [attr2], … [attrN] FROM [table] WHERE [matching condition attr1 … attrN] GROUP BY [attr1], [attr2], … [attrN] ORDER BY `Frequency` DESC
```

Journal of Theoretical and Applied Information Technology
20th February 2015. Vol.72 No.2
© 2005 - 2015 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 4 (page 178) shows a partial table from an example database.

Case 1. "User would like to search subject Programming is usually taught by who at which day". The SQL is as the following:

```
SELECT count(*) as `Frequency`, Lecturer, Day
FROM timetable WHERE Subject= "Programming" GROUP BY Lecturer, Day
ORDER BY `Frequency` DESC
```

The result would be shown as in Table 5:

*Table 5: Results from frequency grouping for case 1.*

| Frequency | Lecturer | Day |
|---|---|---|
| 2 | Ali | Mon |
| 1 | Ali | Tue |
| 1 | Michael | Mon |
| 1 | Fahrul | Mon |

The result shows that Programming is usually taught by Ali on Monday. The same method can continue nesting by searching which Time and Room that Ali most frequently used on Monday. And the result will create a new target case.

Case 2. "User would like to search classroom Room 101 is usually used by who (Lecturer) at what Time". The SQL is as the following:

```
SELECT count(*) as `Frequency`, Lecturer, Time FROM timetable WHERE Classroom= "Room 101" GROUP BY Lecturer, Time
ORDER BY `Frequency` DESC
```

The result would be shown as in Table 6:

*Table 6: Results from frequency grouping for case 2.*

| Frequency | Lecturer | Time |
|---|---|---|
| 3 | Michael | 0800 |
| 1 | Siti | 1300 |

If a user doesn't set any target case for Room 101 to be used by who (Lecturer), then this result would be able to help user identify that Michael is the Lecturer that most frequently use Room 101 at morning 8AM.

### 4.3 Euclidian Distance Similarity Measurement (Value distance measurement)

Euclidian Distance measure the similarity of cases by looking into each attribute's values instead of mere comparison for exact match. There're cases that none of the case in the database able to match the target case, and such situation will return all 0 value similarity if there're no values distance comparison. Euclidian Distance expression as below:

$$\mathbf{SIM(x,y) = 1 - DIST(x,y) = 1 - \sum_i wi\ dist(xi, yi)}$$

$w_i$ stands for the weight of the distance which is similar to priority of each attribute. The total distance would be summing up all differences for every attributes by multiplying them with the weight. Let's give a completely different value as 1, so 1 – sum of all differences is equal to similarity. The higher a similarity value (SIM) indicates the closer it is to target case. Highest SIM value's case will be selected to be the target case. Table 7 shows the cases with 5 attributes each has different weight.

The objective in Table 7 is to search for a case to match for subject "Programming", hence attribute 'Subject' has no weight and is negligible. The attributes take into consideration for similarity are four attributes. Lecturer's proficient in that subject takes the highest priority in measurement thus giving a weight of 3. Day of the class given is considered secondary importance thus giving a weight of 2. Time and Classroom are both equally less important hence giving a weight of 1.

All the cases (Case A, Case B, and Case C) are completely different with the target case (which means no attribute has the exact match). Thus, by using exactly value comparison similarity measurement will return equal values for all 3 cases. By using Euclidian Distance:

Case A: |(1.0 – 0.95) x 3| + |(0.143 – 0.286) x 2| + |(0.33 – 0.708) x 1| + |(0.2 – 0.42) x 1| = 1.034

Case B: |(1.0 – 0.38) x 3| + |(0.143 – 0.429) x 2| + |(0.33 – 0.375) x 1| + |(0.2 – 0.27) x 1| = 2.531

Case C: |(1.0 – 0.7) x 3| + |(0.143 – 0.571) x 2| + |(0.33 – 0.375) x 1| + |(0.2 – 0.62) x 1| = 2.221

All values are above 1.0 thus we'll shift the values by one additional decimal place. Below

Journal of Theoretical and Applied Information Technology
20th February 2015. Vol.72 No.2
© 2005 - 2015 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

are the finalized DIFF values. Case A: 0.1034, Case B: 0.2531, Case C: 0.2221. Thus, the similarity values are as below:

Case A: 1 – 0.1034 = 0.9897

Case B: 1 – 0.2531 = 0.7469

Case C: 1 – 0.2221 = 0.7779

By using Euclidian Distance, the most similar case with the target case is Case A with a similarity value of 0.9897.

### 4.4 Combination of Prioritized Attributes, Frequency Grouping, and Value Difference Measurement

Value from difference measurement (Euclidian Distance) can combine with both Prioritized Attributes and Frequency Grouping.

a. Combine Prioritized Attributes and Euclidian Distance as **COM1** (known requirement)

**SIM(x, y) = 1 – SUM($w_i$ * dist($x_i$ , $y_i$)) + prio_bonus_rules($x$, $y$)**

b. Combine Frequency Grouping and Euclidian Distance as **COM2** (unknown requirement)

**SELECT count(*) as `Frequency`, value - [attr1] as `f1`, value - [attr2] as `f2`, … value - [attrN] as `fN` FROM [table] WHERE [condition attr1 … attrN] GROUP BY `f1`, `f2`, … `fN` ORDER BY `Frequency` DESC**

c. Combine **COM1** and **COM2** (most accurate with more complexity)

If user input 'Target Case', then **COM1** is used If user does not input 'Target Case', then **COM2** is used If **COM1** is used and no result is found, then **COM2** is used.

Thus, with the combination of three concepts from *Prioritized Attributes, Frequency Grouping, and Value Difference Measurement,* the algorithm is shown as the following.

*Table 8: Denotation of Algorithm*

| | |
|---|---|
| $t \in T$ | $T$ is the Universe set of all single target case $t$. |
| $t[c, l, d, s]$ | element **Course, Lecturer, Day, Time** for a single target case $t$. |
| $p \in P$ | $P$ is the Universe set of all single previous case $p$. |
| $p[c, l, d, s]$ | element **Course, Lecturer, Day, Time** for a single previous case $p$. |
| $n \in N$ | $N$ is the Universe set of all single new case $n$. |
| $n[c, l, d, s]$ | element **Course, Lecturer, Day, Time** for a single new case $n$. |
| $i$ | $i$ indicates the indices of element $c$, $l$, $d$, and $s$ respectively |
| pr($x$, $y$) | $x$ as first element of case $p$, $y$ as second element of case $p$, pr($x$, $y$) as given priority on combined matched elements. |
| $U$ | Open set for distance comparison |
| $w$ | weight of an element |
| dist($x$, $y$) | $x$ as $i$-th element of case $t$, $y$ as $i$-th element of case $p$ |

*Algorithm 1: Human Preference Adaptable Retrieval Approach (HPARA)*

```
1.  :BEGIN
2.   U ← ∅
3.   For all  t ∈ T  do
4.     If   t[c],  t[l],  t[d],  t[s]   are all given
5.       For  all   p ∈ P  do
6.          U.Insert(p, 1 −  SUM(w_i ×
             dist(t[i]  , p[i])) +  pr(i , j))
7.       /* Optional: Loop u ← U. PopTopScore ( )
          if does not meet Soft Constraint */
8.      If  U.TopScore( ) = 0   then
9.      HPARA_SecondPhase( t )   (Algorithm 2)
10.    Else
11.       Clash ← False
12.       Do
13.          u ← U.PopTopScore( )
14.          For   all   n ∈ N  do
15.             If  n = u  then
16.                Clash ← True
17.       While  Clash = True  AND  NOT  U  is
          Empty
18.       If  Clash = True  OR  U is Empty  then
19.          HPARA_SecondPhase( t )
20.       Else
21.          N.Insert(u)
22.   Else
23.     If NOT  t[c]  =  NULL  then
24.       HPARA_SecondPhase( t )
25.  :END
```

Journal of Theoretical and Applied Information Technology
20th February 2015. Vol.72 No.2
© 2005 - 2015 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

*Algorithm 2: Human Preference Adaptable Retrieval Approach (HPARA) SecondPhase(t)*

```
1. :BEGIN
2.  If  NOT  t[l]  =  NULL  then
3.    u ← RetrieveCBR ( t[c], t[l] )  (Algorithm 3)
4.    If NOT  u  =  NULL  then
5.       N.Insert(u)
6.    Else
7.       u ← RetrieveCBR ( t[c], NULL )
8.       If NOT  u  =  NULL  then
9.          N.Insert(u)
10.      Else
11.         N.Insert( Random  Case )
12. Else
13.   u ← RetrieveCBR ( t[c], NULL )
14.   If NOT  u  =  NULL  then
15.      N.Insert(u)
16.   Else
17.      N.Insert( Random  Case )
18. :END
```

Algorithm 1 shows the pseudo code of HPARA with the process flow start from capture input from all single target case *t*. The algorithm then check to confirm all elements (course, lecturer, day, and time) needed for the target case was provided, if not process will continue with Algorithm 2 with at least course is provided. If all elements are provided, process continues with comparing distance for target case and previous case using *Euclidian Distance Similarity Measurement* including the matched bonus from the concept of *Linked Prioritized Attributes.* Retrieval process will continue with Algorithm 2 if best distance score equal to 0, or else, all single new case will check for clash status with the highest distance score case and if clash status is negative, case will be retrieve.

For any condition that falls into Algorithm 2, which shows the pseudo code of HPARA second phase, process flow will check target case with both elements (course, lecturer) using Algorithm 3 (Case-based reasoning retrieval) to search for similar past cases, if no result return, process will then check target case again with only element 'course' for suitable case. However, if still no result return with only 'course' element, process will return a random case.

The *Frequency Grouping* concept was apply in Algorithm 3 using Case-based reasoning approach which will check for the highest frequency past cases with the desired combination of elements. Cases will retrieve if clash status is negative.

*Algorithm 3: Case-based reasoning retrieval RetrieveCBR ( t[c], t[l] )*

```
1. :BEGIN
2.  U ← ∅
3.  If  NOT  t[l]  =  NULL  then
4.     For  all  p ∈P  do
5.        U.Insert(p, Frequency(t[l], t[d], t[s]))
6.  Else
7.     For  all  p ∈P  do
8.        U.Insert(p, Frequency(t[d], t[s]))
9.  u ← U.PopTopFrequency( )
10. /* Optional: Loop u ← U.PopTopFrequency( )
if does not
    meet Soft Constraint */
11. Clash ← False
12. For  all  n ∈N  do
13.   If  n = u  then
14.      Clash ← True
15. If  Clash = False  then
16.   Return  u
17. Else
18.   Return NULL
19. END
```

## 5. EXPERIMENT DESIGN

The experiment is performed in 2 different environments with total in 4 phases which is 2 phases in each environment. The first environment in the experiment will be running by generating the time table with the increment of number of components against times taken to generate the time table. Number of components will be increase by 3 each time until over 100 components is selected to generate time table. Elapsed time is taken each time components were increased. The purpose of this experiment phase aim to compare the elapsed time taken to generate a time table for HPARA and GA algorithm in both conditions; with and without fulfill soft constraint capability. The outcomes from this phase expect to prove the effectiveness of HPARA algorithm with lower *elapsed time* taken to generate time table. The reason for the first environment to separate into 2 phases; with and without fulfill soft constraint is important to test the handling capability for HPARA and GA algorithm. Solving a time tabling problem without considering or fulfilling the soft constraint is obviously less complex. However, the

© 2005 - 2015 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

test on both algorithms required to run in a less complex and more complex phase in order to get an accurate result for handling time, and to compare the difference.

The second environment of the experiment will be running with the increment of number of components against number of soft constraint violations. Same as the first environment, the experiment is divided into 2 phases; with and without fulfill soft constraint capability for both HPARA and GA algorithms. The importance in the experiment is to test how good one algorithm able to meet requirements (soft constraint). The outcomes of this phase expect to prove the accuracy of HPARA algorithm with lower numbers of *soft constraint violation* in both phases; with and without fulfill soft constraint capability algorithm.

## 6. EXPERIMENTAL RESULTS AND DISCUSSION

The first experiment demonstrates the time taken to generate a schedule with the comparison of Genetic Algorithm and Case-based reasoning both without including soft constraint and including soft constraint to prove the effectiveness of Case-based reasoning in generate a new schedule.

### 6.1 Elapsed time for generating new schedule comparison

*Table 9: Elapsed time for generate new schedule (without soft constraint*

| Without Soft Constraints | | Elapsed Time (millisecond) | |
|---|---|---|---|
| | | GA | CBR |
| No. Of Components | 3 | 63 | 114 |
| | 6 | 95.6 | 25.2 |
| | 9 | 156.2 | 31.8 |
| | 12 | 205.4 | 39.4 |
| | 15 | 271 | 47 |
| | 18 | 341.2 | 56 |
| | 21 | 419.2 | 68.8 |
| | 24 | 493.4 | 76.8 |
| | 27 | 584 | 81.2 |
| | 30 | 693.2 | 104.2 |
| | 33 | 781.6 | 107.4 |
| | 36 | 895.2 | 110 |
| | 39 | 1002 | 124.4 |
| | 42 | 1115 | 121.8 |
| | 45 | 1244.8 | 124.2 |
| | 48 | 1387.8 | 124 |
| | 51 | 1525 | 139.6 |
| | 54 | 1699.4 | 151.8 |
| | 57 | 1850 | 150 |
| | 60 | 1993.8 | 159.6 |
| | 63 | 2160.4 | 179.4 |
| | 66 | 2311.2 | 180.2 |
| | 69 | 2496 | 183.4 |
| | 72 | 2687 | 198.2 |
| | 75 | 2884.6 | 218.4 |
| | 78 | 3066.6 | 213.2 |
| | 81 | 3252 | 233.4 |
| | 84 | 3463.4 | 233.8 |
| | 87 | 3701.6 | 244.6 |
| | 90 | 3905.8 | 261.2 |
| | 93 | 4148 | 269.6 |
| | 96 | 4399.4 | 277.8 |
| | 99 | 4594.8 | 288.8 |
| | 102 | 4902.4 | 298 |

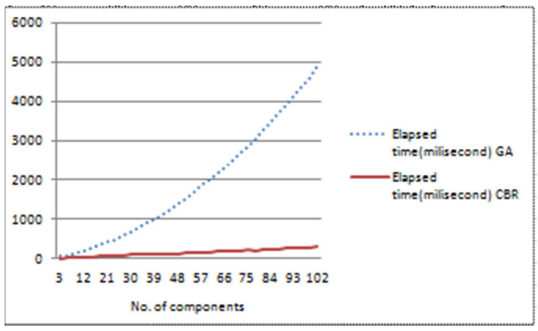


*Figure 6: Elapsed time to generate new schedule base on number of components (without fulfill soft constrain)*

Journal of Theoretical and Applied Information Technology
20th February 2015. Vol.72 No.2
© 2005 - 2015 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

*Table 10. Elapsed time for generate new schedule (with soft constraint)*

| With Soft Constraints | | Elapsed Time (millisecond) | |
|---|---|---|---|
| | | GA | CBR |
| No. Of Components | 3 | 63 | 114 |
| | 6 | 95.6 | 25.2 |
| | 9 | 156.2 | 31.8 |
| | 12 | 205.4 | 39.4 |
| | 15 | 271 | 47 |
| | 18 | 341.2 | 56 |
| | 21 | 419.2 | 68.8 |
| | 24 | 493.4 | 76.8 |
| | 27 | 584 | 81.2 |
| | 30 | 693.2 | 104.2 |
| | 33 | 781.6 | 107.4 |
| | 36 | 895.2 | 110 |
| | 39 | 1002 | 124.4 |
| | 42 | 1115 | 121.8 |
| | 45 | 1244.8 | 124.2 |
| | 48 | 1387.8 | 124 |
| | 51 | 1525 | 139.6 |
| | 54 | 1699.4 | 151.8 |
| | 57 | 1850 | 150 |
| | 60 | 1993.8 | 159.6 |
| | 63 | 2160.4 | 179.4 |
| | 66 | 2311.2 | 180.2 |
| | 69 | 2496 | 183.4 |
| | 72 | 2687 | 198.2 |
| | 75 | 2884.6 | 218.4 |
| | 78 | 3066.6 | 213.2 |
| | 81 | 3252 | 233.4 |
| | 84 | 3463.4 | 233.8 |
| | 87 | 3701.6 | 244.6 |
| | 90 | 3905.8 | 261.2 |
| | 93 | 4148 | 269.6 |
| | 96 | 4399.4 | 277.8 |
| | 99 | 4594.8 | 288.8 |
| | 102 | 4902.4 | 298 |

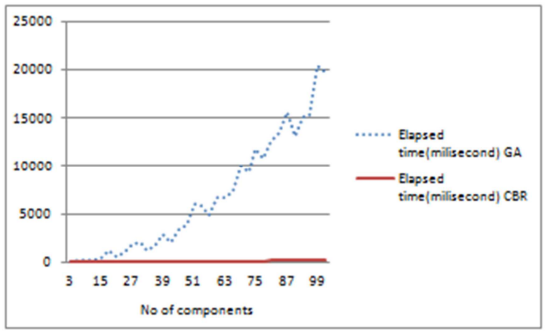


*Figure 7: Elapsed time to generate new schedule base on number of components (with fulfill soft constrain)*

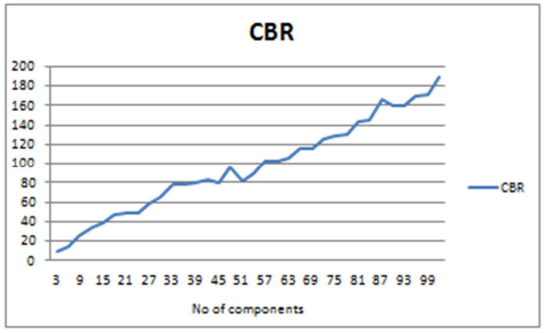


*Figure 8: Elapsed time for CBR to generate schedule based on increment of components*

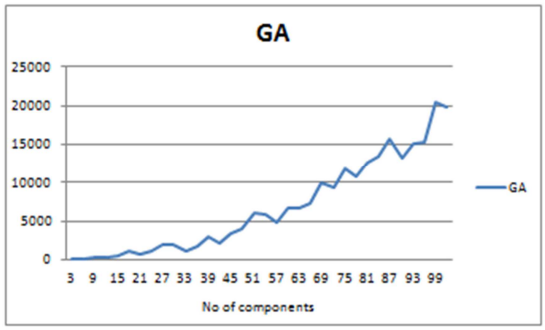


*Figure 9: Elapsed time for GA to generate schedule based on increment of components.*

Experiments results from Table 8 and Table 9 was the average readings of time taken to generate a new schedule. With every increment of 3 components, experiment was run 5 times to get an average of 5 readings for both GA and CBR. Results in Figure 6 shown that elapsed time for GA is increasing exponentially. Compare to CBR, with each schedule generate with more components, time taken was only slightly increase in linear.

Journal of Theoretical and Applied Information Technology
20th February 2015. Vol.72 No.2
© 2005 - 2015 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

For running both experiments to fulfill soft constraint and without, results clearly shown that CBR will be able to generate a new schedule in the average time of 200ms with around 100 components, while GA elapsed time to generate a new schedule had been increase from average 5,000ms (without fulfill soft constraint) to 25,000ms (fulfill soft constraint).

### 6.2 Soft constraint violation comparison

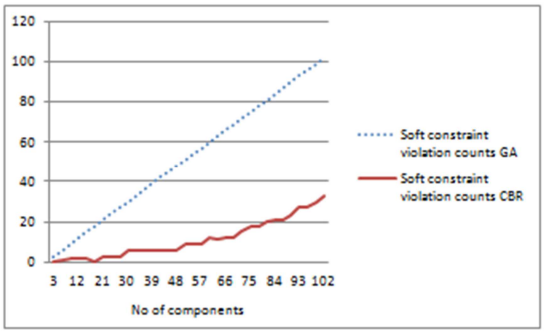


*Figure 10.: Soft constraint violation counts for GA and CBR (without using soft constraint fulfillment algorithm)*

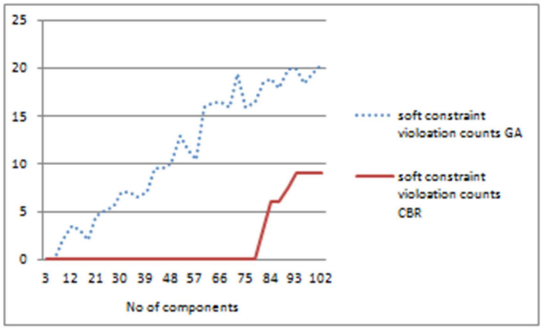


*Figure 11.: Soft constraint violation counts for GA and CBR (using soft constraint fulfillment algorithm)*

The second experiment is run to compare the accuracy of both algorithm capabilities to fulfill soft constraint while new schedule is plot. Figure 10 shows the results of using algorithm *without* soft constraint check for both GA and CBR. Figure 11 shows the results when using algorithm *with* soft constraint check.

From both readings, we can see the distinct different comes from CBR as even using an algorithm that without soft constraint check, violations of soft constraint counts were at most 30% from plotting 100 components. For further confirmation while experiment is running with the finalize algorithm which will run soft constraint check, CBR will only start to have soft constraint violation start from plotting 80 components and above.

## 7. CONCLUSION AND FUTURE WORK

The HPARA approach in Case-based reasoning had been evaluated through the experiments. The method is being evaluated using past datasets from Faculty of Computer Science and Information Technology, University Putra Malaysia. By obtaining the lower counts of soft constraint violation, it proves the performance for accuracy while generating a new schedule. The results of the proposed method showed there will be no soft constraint violation with lower amount of components. In the mean time, Time taken results is making great improvements for schedule plotting with the rating of average 200ms. Experiments had been shown that HPARA approach has successfully achieve in both accuracy as well as reduced time taken in solving scheduling in University time tabling. However, comparing to the retrieval method was used in this paper, which is component by component retrieval; future direction may further improve into a full case retrieval, converting the case components into certain similarity points for requirements matching. This may further decrease the retrieval processing time.

## REFERENCES

[1] Rhydian Lewis: A Survey of Metaheuristic-based Techniques for University Timetabling Problems, *OR Spectrum (2007), 30(1),* 167–190.

[2] Pádraig Cunningham, Barry Smyth: Case-Based Reasoning in Scheduling: Reusing Solution Components. The International Journal of Production Research. (1997) 35: 2947-2961.

[3] Beligiannis, G. N., Moschopoulos, C. N., Kaperonis, G. P., & Likothanassis, S. D.: Applying evolutionary computation to the school timetabling problem: The Greek case. Computers and Operations Research (2008), 35(4), 1265–1280.

[4] Beligiannis, G. N., Moschopoulos, C. N., & Likothanassis, S. D.: A genetic algorithm approach to school timetabling. Journal of the Operational Research Society (2009), 60(1), 23–42.

[5] Ross, P., Hart, E., & Corne, D.: Genetic algorithms and timetabling natural computing

© 2005 - 2015 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

series. Advances in Evolutionary Computing: Theory and Applications . (2003), 755–777.

[6] Nedjah, N., de Macedo Mourelle, L.: Evolutionary Pattern Matching Using Genetic Programming Studies in Computational Intelligence (SCI) 13 (2006), 81-104.

[7] Qarouni Fard, D., Ferdowsi Univl, Mashad, Najafi-Ardabili, A; Moeinzadeh, M.-H.: Finding Feasible Timetables with Particle Swarm Optimization. Innovations in Information Technology, 2007. IIT '07. 4th International Conference, 2007

[8] Burke, E. K., Kendall, G., & Soubeiga, E.: A tabu-search hyperheuristic for timetabling and rostering. Journal of Heuristics, 9(6) (2003), 451–470.

[9] Di Gaspero, L., Schaerf, A.: Tabu search techniques for examination timetabling. Lecture Notes in Computer Science Vol. 2079 (2001), 104–17.

[10] Socha, K., Knowles, J., Sampels, M.: A MAX–MIN ant system for the university course timetabling problem. Lecture Notes in Computer Science Vol. 2463, Springer, Berlin (2002), 1– 13.

[11] Burke, E.K. Maccarthy, B.L. Petrovic, S. and Qu, R: Case-based Reasoning in Course Timetabling: An Attribute Graph Approach. Case-Based Reasoning Research and Development Lecture Notes in Computer Science Volume 2080, Springer, Berlin (2001), 90-104.

[12] A. Aamodt, E. Plaza: Case-based reasoning: foundational issues, methodological variations, and system approaches, Artificial Intelligence Communications 7 (1994), 39–59.

[13] Kolodner, J: Maintaining organisation in a dynamic long-term memory. Cognitive Science, 7 (1983), 234–280.

[14] J. Hunt: Evolutionary case based design, in: I.D. Waston (Ed.) Progress in Case-based Reasoning, LNAI 1020, Springer, Berlin (1995), 17–31.

[15] Koton P, SMARTlan: A case-based resource allocation and scheduling system, in: Proceedings: Workshop on Case-based Reasoning (DARPA) (1989), 285-289.

[16] Bezirgan A, A case-based approach to scheduling constraints, in: Dorn J and Froeschl KA ed.,Scheduling of Production Processes, (Ellis Horwood Limited, 1993) 48-60.

[17] Miyashita K and Sycara K, Adaptive case-based control of scheduling revision, in: Zweben M and Fox MS, eds., Intelligent Scheduling, (Morgan Kaufmann, 1994) 291-308.

[18] Schmidt G, Case-based reasoning for production scheduling, International Journal of Production Economics 56-57 (1998) 537-546.

[19] E. K. Burke, B. MacCarthy, S. Petrovic, R. Qu: Structured Cases in CBR – Re-using and Adapting Cases for Timetabling Problems, Knowledge-Based Systems, 13(2-3) (2000), 159-165.

[20] Burke, E.K. Maccarthy, B.L. Petrovic, S. and Qu, R.: Multiple-retrieval case-based reasoning system for course timetabling problems. Journal of Operations Research Society (2005).

[21] Burke, E.K,. Petrovic, S. and Qu, R. Case-based heuristic selection for timetabling problems. Journal of Scheduling (2006), vol 9, no. 2, 115-132.

[22] Watson, I., & Marir, F.: Case-based reasoning: a review. The Knowledge Engineering Review, 9(4) (1994), 327–354.

[23] Rossi-Doria, O., Sampels, M, Birattari, M., Chiarandini, M., Dorigo, M., Gambardella, L. M., et al.: A comparison of the performance of different metaheuristics on the timetabling problem. Lecture Notes In Computer Science Vol. 2740 (2003), (pp. 329–351). Berlin: Springer.

[24] Mohammed Azmi Al-Betar, Ahamad Tajudin Khader, Member, IEEE, and Munir Zaman, Member, IEEE: University Course Timetabling Using a Hybrid Harmony Search Metaheuristic Algorithm. IEEE Transactions On Systems, Man, And Cybernetics—Part C: Applications And Reviews (2012), Vol. 42, No. 5.

[25] Shengxiang Yang, Member, IEEE, and Sadaf Naseem Jat: Genetic Algorithms With Guided and Local Search Strategies for University Course Timetabling. IEEE Transactions On Systems, Man, And Cybernetics—Part C: Applications And Reviews (2011), Vol. 42, No. 5.

[26] Chuanmin Mi, Hanchong Qian, Sifeng Liu, *Member, IEEE*, Zhansheng Chang: Study on Case Retrieving in Case-based Reasoning Based on Grey Incidence Theory and Its Application in Bank Regulation. IEEE International Conference on Fuzzy Systems (2008).

[27] L.B. Romdhane & B. Ayeb: An evolutionary algorithm for abductive reasoning, Journal of Experimental & Theoretical Artificial Intelligence (2011), 23:4, 529-544.

© 2005 - 2015 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[28] Susan Fox & David B. Leake: Introspective reasoning for index refinement in case-based reasoning, Journal of Experimental &Theoretical Artificial Intelligence (2001), 13:1, 63-88.

[29] Gavin Finnie, Zhaohao Sun: R 5 model for case-based reasoning, Knowledge-Based Systems 16 (2002), 59–65.

[30] D.B. Leake: Case-Based Reasoning: Experiences, Lessons and Future Direction, AAAI Press/MIT Press, Menlo Park, CA (1996).

© 2005 - 2015 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

*Table 4: Example Database Cases*

| | Subject | Lecturer | Day | Time | Classroom |
|---|---|---|---|---|---|
| **Case A** | Programming | Ali | Tue | 1700 | Room 203 |
| **Case B** | Programming | Michael | Mon | 0800 | Room 101 |
| **Case C** | Programming | Ali | Mon | 0900 | Room 304 |
| **Case D** | Multimedia | Michael | Wed | 0800 | Room 101 |
| **Case E** | Multimedia | Siti | Thur | 1300 | Room 203 |
| **Case F** | Programming | Ali | Mon | 0900 | Room 304 |
| **Case G** | Programming | Fahrul | Mon | 1700 | Room 304 |
| **Case H** | Database | Siti | Thur | 1300 | Room 101 |
| **Case I** | Multimedia | Michael | Tue | 0800 | Room 101 |
| **Case J** | Database | Ali | Wed | 1700 | Room 203 |
| **Case K** | Database | Fahrul | Fri | 0800 | Room 304 |

*Table 7: Cases with weight.*

| | | **Max: 1, Min: 0** | **Max: 7, Min: 1** | **Max: 24, Min: 0** | **Max: 5.5, Min: 1.1** |
|---|---|---|---|---|---|
| | **Subject** | **Lecturer (w=3)** | **Day (w=2)** | **Time (w=1)** | **Classroom (w=1)** |
| **Target Case** | Programming | Ali (1.0) | Mon (0.48) | 0800 (0.38) | Room 101 (0.2) |
| **Case A** | Programming | Siti (0.95) | Tue (0.286) | 1700 (0.708) | Room 203 (0.42) |
| **Case B** | Programming | Michael (0.38) | Wed (0.429) | 0900 (0.375) | Room 105 (0.27) |
| **Case C** | Programming | Lee (0.7) | Thur (0.571) | 0900 (0.375) | Room 304 (0.62) |