A Divide-and-Conquer Tabu Search Approach for Online Test Paper Generation

Minh Luan Nguyen[1], Siu Cheung Hui[1], and Alvis C.M. Fong[2]

[1] Nanyang Technological University, Singapore
{NGUY0093,asschui}@ntu.edu.sg
[2] Auckland University of Technology, New Zealand
acmfong@gmail.com

Abstract. Online Test Paper Generation (Online-TPG) is a promising approach for Web-based testing and intelligent tutoring. It generates a test paper automatically online according to user specification based on multiple assessment criteria, and the generated test paper can then be attempted over the Web by user for self-assessment. Online-TPG is challenging as it is a multi-objective optimization problem on constraint satisfaction that is NP-hard, and it is also required to satisfy the online runtime requirement. The current techniques such as dynamic programming, tabu search, swarm intelligence and biologically inspired algorithms are ineffective for Online-TPG as these techniques generally require long runtime for generating good quality test papers. In this paper, we propose an efficient approach, called DAC-TS, which is based on the principle of constraint-based divide-and-conquer (DAC) and tabu search (TS) for constraint decomposition and multi-objective optimization for Online-TPG. Our empirical performance results have shown that the proposed DAC-TS approach has outperformed other techniques in terms of runtime and paper quality.

Keywords: Online test paper generation, multi-objective optimization, web-based testing, intelligent tutoring system.

1 Introduction

With the rapid growth of E-learning, Web-based testing and intelligent tutoring [2, 5, 13] have become popular for self-assessment and learning in an educational environment. To support Web-based testing and intelligent tutoring, Online Test Paper Generation (Online-TPG) is a promising approach which generates a test paper automatically online according to user specification based on multiple assessment criteria, and the generated test paper can then be attempted over the Web by user. More specifically, Online-TPG aims to find an optimal subset of questions from a question database to form a test paper based on criteria such as total time, topic distribution, difficulty degree, discrimination degree, etc.

Online-TPG is a challenging problem. Firstly, TPG is categorized as a multi-objective optimization problem on constraint satisfaction which is NP-hard [10]. Secondly, the current TPG techniques [6–10, 12, 16] have not taken the online generation requirement into consideration as TPG is traditionally considered as an offline process similar to other multi-objective optimization problems such as timetabling and job-shop scheduling [4].

D. Wang and M. Reynolds (Eds.): AI 2011, LNAI 7106, pp. 717–726, 2011.
© Springer-Verlag Berlin Heidelberg 2011

These current techniques have optimized an objective function based on multi-criteria constraints and weighting parameters for test paper quality. However, determining appropriate weighting parameters is quite difficult and computationally expensive. And these techniques generally require long runtime for generating good quality test papers.

In this paper, we propose an efficient approach, called DAC-TS, which is based on the principle of constraint-based divide-and-conquer (DAC) and tabu search (TS) for Online-TPG. The rest of the paper is organized as follows. Section 2 reviews the related techniques for automatic test paper generation. Section 3 gives the problem specification. The proposed DAC-TS approach for Online-TPG is presented in Section 4. Section 5 gives the performance evaluation of the proposed approach and its comparison with other TPG techniques. Finally, Section 6 gives the conclusion.

2 Related Work

In [10], tabu search (TS) was proposed to construct test papers by defining an objective function based on multi-criteria constraints and weighting parameters for test paper quality. TS optimized test paper quality by the evaluation of the objective function. In [8], dynamic programming optimized an objective function incrementally based on the recursive optimal relation of the objective function. In [9], a genetic algorithm (GA) was proposed to generate quality test papers by optimizing a fitness ranking function based on the principle of population evolution. In [16], differential evolution (DE) was proposed for test paper generation. DE is similar to the spirit of GA with some modifications on solution representation, fitness ranking function, and the crossover and mutation operations to improve the performance. In [12], an artificial immune system (AIS) was proposed to use the clonal selection principle to deal with the highly similar antibodies for elitist selection in order to maintain the best test papers for different generations.

In addition, swarm intelligence algorithms such as particle swarm optimization and ant colony optimization have also been investigated for TPG. In [6], particle swarm optimization (PSO) was proposed to generate multiple test papers by optimizing a fitness function which is defined based on multi-criteria constraints. In [7], ant colony optimization (ACO) was proposed to generate quality test papers by optimizing an objective function which is based on the simulation of the foraging behavior of real ants.

3 Problem Specification

Let $\mathcal{Q} = \{q_1, q_2, .., q_n\}$ be a dataset consisting of n questions, $\mathcal{C} = \{c_1, c_2, .., c_m\}$ be a set of m topics, and $\mathcal{Y} = \{y_1, y_2, .., y_k\}$ be a set of k question types such as multiple choice questions, fill-in-the-blanks and long questions. Each question $q_i \in \mathcal{Q}$, where $i \in \{1, 2, .., n\}$, has 8 attributes $\mathcal{A} = \{q, o, a, e, t, d, c, y\}$, where q is the question identity, o is the question content, a is the question answer, e is the discrimination degree, t is the question time, d is the difficulty degree, c is the related topic and y is the question type. Table 1 shows a sample Math question dataset.

A *test paper specification* $\mathcal{S} = \langle N, T, D, C, Y\rangle$ is a tuple of 5 attributes which are defined based on the attributes of the selected questions as follows: N is the number of questions, T is the total time, D is the average difficulty degree, $C = \{(c_1, pc_1),.., (c_M,$

Table 1. An Example of Math Dataset

(a) Question Table

$\mathcal{Q}$_ID	o	a	e	t	d	c	y
q_1	...	...	4	8	1	c_1	y_2
q_2	...	...	7	9	2	c_1	y_2
q_3	...	...	4	6	6	c_2	y_1
q_4	...	...	5	9	9	c_2	y_2
q_5	...	...	4	7	4	c_1	y_1
q_6	...	...	7	4	7	c_2	y_1

(b) Topic Table

$\mathcal{C}$_ID	$name$
c_1	Integration
c_2	Differentiation

(c) Question Type Table

$\mathcal{Y}$_ID	$name$
y_1	Multiple choice
y_2	Fill-in-the-blank

$pc_M)\}$ is the specified proportion for topic distribution and $Y = \{(y_1, py_1),.., ((y_K, py_K)\}$ is the specified proportion for question type distribution.

The test paper generation process aims to find a subset of questions from a question dataset $\mathcal{Q} = \{q_1, q_2, .., q_n\}$ to form a test paper P with specification $\mathcal{S}_P$ that maximizes the average discrimination degree and satisfies the test paper specification such that $\mathcal{S}_P = \mathcal{S}$. It is important to note that the test paper generation process occurs over the Web where user expects to generate a test paper within an acceptable response time. Therefore, Online-TPG is as hard as other optimization problems due to its computational NP-hardness, and it is also required to be solved efficiently in runtime.

4 Proposed Approach

In this paper, we propose a constraint-based Divide-And-Conquer Tabu Search (DAC-TS) approach for Online-TPG. As the constraints specified in the test paper specification can be formulated as a standard 0-1 fractional Integer Linear Programming (ILP) problem [10] in the form of linear equality constraints, we can decompose the constraints into two independent subsets, namely *content constraints* and *assessment constraints*, which can then be solved separately and progressively. In the test paper specification $\mathcal{S} = \langle N, T, D, C, Y\rangle$, the content constraints include the constraints on topic distribution C and question type distribution Y, whereas the assessment constraints include the constraints on total time T and average difficulty degree D.

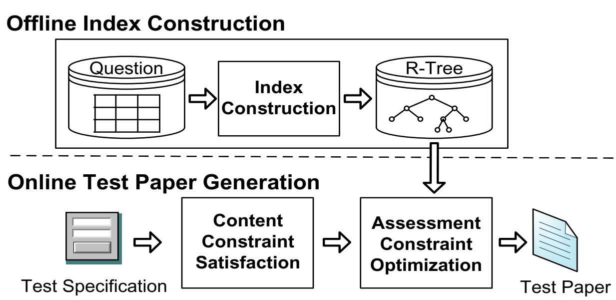


Fig. 1. The Proposed DAC-TS Approach

The proposed DAC-TS approach, as shown in Figure 1, consists of 2 main processes: Offline Index Construction and Online Test Paper Generation. In the Offline Index Construction process, it constructs an effective indexing structure for supporting tabu search to improve the quality of the generated paper. In the Online Test Paper Generation process, it generates a high quality test paper that satisfies the specified content constraints and assessment constraints. As illustrated in Figure 1, it consists of 2 major steps: Content Constraint Satisfaction and Assessment Constraint Optimization.

4.1 R-Tree Index Construction

We propose to use an effective 2-dimensional data structure, called R-Tree, to store questions based on the time and difficulty degree attributes. R-Tree has been widely used for processing queries on 2-dimensional spatial databases. As there is no specified rule on grouping of data into nodes in R-Tree, different versions of R-tree have been proposed [1, 14]. The R-Tree used here is similar to the R-tree version discussed in [1], with some modifications on index construction in order to enhance the efficiency. Some of the modified operations include insertion, subtree selection, overflow handling, and node splitting for index construction. Each leaf node in a R-Tree is a Minimum Bounding Rectangle (MBR) which is the smallest rectangle in the spatial representation that tightly encloses all data points located in the leaf node. Each non-leaf node has child nodes, which contain MBRs at the lower level. Figure 2 illustrates the R-Tree constructed from the Math dataset.

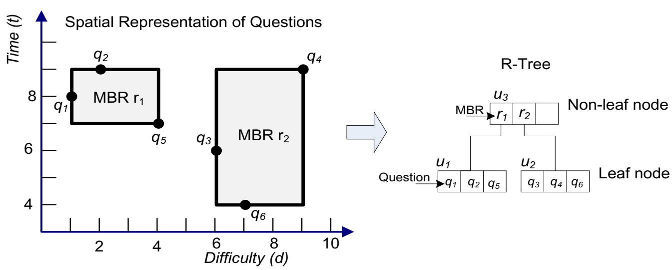


Fig. 2. An Example R-Tree

4.2 Content Constraint Satisfaction

It is quite straightforward to generate an initial test paper that satisfies the content constraints based on the number of questions N. Specifically, the number of questions of each topic c_l is $pc_l * N$, $l = 1..M$. Similarly, the number of questions of each question type y_j is $py_j * N$, $j = 1..K$. There are several ways to assign the N pairs of topic-question type to satisfy the content constraints. Here, we have devised an approach which applies a heuristic to try to achieve the specified total time early. To satisfy the content constraints, the round-robin technique is used for question selection. More specifically, for each topic c_l, $l = 1..M$, we assign questions alternately with various question types y_j, $j = 1..K$, as much as possible according to the number of questions.

Then, for each of the N pairs of topic-question type (c_l, y_j) obtained from the round-robin selection step, we assign a question q from the corresponding topic-question type (c_l, y_j) that has the highest question time to satisfy the total time early.

4.3 Assessment Constraint Optimization

Assessment Constraint Violation indicates the differences between the test paper specification and the generated test paper according to the total time constraint $\triangle T(\mathcal{S}_P, \mathcal{S})$ and the average difficulty degree constraint $\triangle D(\mathcal{S}_P, \mathcal{S})$ as follows:

$$\triangle T(\mathcal{S}_P, \mathcal{S}) = \frac{|T_P - T|}{T} \quad \text{and} \quad \triangle D(\mathcal{S}_P, \mathcal{S}) = \frac{|D_P - D|}{D}$$

A generated test paper P with specification $\mathcal{S}_P = \langle N, T_P, D_P, C_P, Y_P \rangle$ is said to satisfy the assessment constraints in $\mathcal{S}$ if $\triangle T(\mathcal{S}_P, \mathcal{S}) \leq \alpha$ and $\triangle D(\mathcal{S}_P, \mathcal{S}) \leq \beta$, where α and β are two predefined thresholds that indicate the acceptable quality satisfaction on total time and average difficulty degree respectively.

In addition, an objective function is defined for evaluating the quality of test papers based on assessment constraint violations. The quality of a generated test paper P is defined by the following objective function:

$$f(P) = \triangle T(\mathcal{S}_P, \mathcal{S})^2 + \triangle D(\mathcal{S}_P, \mathcal{S})^2$$

In Assessment Constraint Optimization, we conduct tabu search to improve the quality of the test paper by minimizing assessment constraint violations. This optimization process is repeated until the termination conditions are reached.

4.4 Tabu Search

Tabu search [3] is an iterative search method, which aims to find better questions to substitute the existing questions in the test paper in order to minimize assessment constraint violation. To form a new test paper, each question q_k in the original test paper P_0 is substituted by another better question q_m which has the same topic and question type such that assessment constraint violations are minimized. The tabu search comprises a local search with 3 strategies: Memory Usage, Up-hill Movement and Memory Relaxation. The termination conditions for the tabu search are based on the quality satisfaction and the maximum number of iterations in which no better test paper can be found.

In *memory usage*, DAC-TS uses a short-term memory and a long-term memory to avoid visiting a solution repeatedly. The *recency-based short-term memory* is used to prevent the substitution of a specific question in the current test paper for some steps after it has just been substituted. This short-term memory, namely TS, is implemented as follows: when a question q_i is substituted, the position i of that question is put into the short-term tabu list TS with a tenure t_{TS}. After each move, the tenure of the current entries in the TS is decreased by 1 and those entries with zero tenure are dropped from the TS. Whereas the *transitional frequency-based long-term memory* is used to dynamically avoid using over-active questions that have a specific topic-question type in order to help diversification and prevent cycling. To achieve this, a Move Frequency Table (MFT) has been incorporated into the tabu search process to store the move frequency of each topic-question type. This long-term memory, namely TL, is implemented as follows: when a question q_i is substituted, the move frequency of the topic-question type of that question is incremented by 1. If an entry x has been moved more than two times and TL is not full, it will be put into TL. If TL is full and some entries y in TL have a lower move frequency than x, we remove y from TL and add x into TL.

In *up-hill movement*, tabu search can accept a move even if the quality of the next solution is worse than that of the current solution. The reason is to escape the local optimal region and explore other new promising regions in the search space. However, to ensure that the up-hill process will not go too far from the current best solution, we set the following condition: $\frac{f(P)-f_{best}}{f_{best}} \leq r$ where r is a predefined threshold, and $f(P)$ and f_{best} are the values of the objective function of the current test paper P and the current best solution respectively.

Finally, *memory relaxation* is used to relax the tabu lists. If a given number of iterations has elapsed and TL is full since the last best solution was found, or if the current solution is much worse than the last best solution, we empty all entries in both TS and TL. Relaxation of the tabu lists will change the neighborhood of the current solution drastically, which may drive the search into a new promising region and increase the likelihood of finding a better solution.

Pruning Search Space. As the neighborhood region is very large, we need to prune the search space to find a 2-dimensional region W that contains possible questions for substitution. Let $\mathcal{S}_{P_0} = \langle N, T_0, D_0, C_0, Y_0\rangle$ be the specification of a test paper P_0 generated from a specification $\mathcal{S} = \langle N, T, D, C, Y\rangle$. Let P_1 be the test paper created after substituting a question q_k of P_0 by another question $q_m \in \mathcal{Q}$ with $\mathcal{S}_{P_1} = \langle N, T_1, D_1, C_1, Y_1\rangle$. The relations of total time and average difficulty degree between P_1 and P_0 can be expressed as follows:

$$T_1 = T_0 + t_m - t_k \tag{1}$$
$$D_1 = D_0 + \frac{d_m}{N} - \frac{d_k}{N} \tag{2}$$

where t_k and t_m are the question time of q_k and q_m respectively, and d_k and d_m are the difficulty degree of q_k and q_m respectively.

Let's consider the total time violation of P_0. If $\triangle T\ (\mathcal{S}_{P_0}, \mathcal{S}) = \frac{|T_0-T|}{T} \geq \alpha$ and $T_0 \leq T$, where α is the predefined threshold. To improve the total time satisfaction of P_1, q_m should have the question time value of $t_k + (T - T_0)$ such that $\triangle T(\mathcal{S}_{P_1}, \mathcal{S})$ is minimized. Furthermore, as $\triangle T(\mathcal{S}_{P_1}, \mathcal{S}) = \frac{|T_1-T|}{T} \leq \alpha$, q_m should have the total time t_m in the interval $t_k + (T - T_0) \pm \alpha T$. Therefore, we have $t_m \in [t_k + T - T_0 - \alpha T$, $t_k + T - T_0 + \alpha T]$. If $\triangle T(\mathcal{S}_{P_0}, \mathcal{S}) = \frac{|T_0-T|}{T} \geq \alpha$ and $T_0 > T$, we can also derive the same result. Similarly, we can derive the result for the difficulty degree of q_m: $d_m \in [d_k + N(D - D_0) - \beta ND$, $d_k + N(D - D_0) + \beta ND]$, where D_0, D and β are the average difficulty degree of P_0 and $\mathcal{S}$, and the predefined threshold respectively.

Finding the Best Question for Substitution. Among all the questions located in the 2-dimensional region W, it finds the best question that minimizes the objective function in order to enhance the test paper quality. Consider question q_m as a pair of variables on its question time t and difficulty degree d. The objective function $f(P_1)$ can be considered as a multivariate function $f(t, d)$:

$$f(P_1) = f(t,d) = \triangle T(\mathcal{S}_{P_1}, \mathcal{S})^2 + \triangle D(\mathcal{S}_{P_1}, \mathcal{S})^2 = (\frac{T_1 - T}{T})^2 + (\frac{D_1 - D}{D})^2$$

From Equations (1) and (2), we have:

$$f(t,d) = \frac{(t - T + T_0 - t_k)^2}{T^2} + \frac{(d - ND + ND_0 - d_k)^2}{D^2} = \frac{(t-t^*)^2}{T^2} + \frac{(d-d^*)^2}{D^2}$$
$$\geq \frac{(t-t^*)^2 + (d-d^*)^2}{T^2 + D^2} = \frac{distance^2(q_m, q^*)}{T^2 + D^2}$$

where q^* is a question having question time $t^* = T - T_0 + t_k$ and difficulty degree $d^* = ND - ND_0 + d_k$.

As T and D are predefined constants and q^* is a fixed point in the 2-dimensional space, the good question q_m to replace question q_k in P_0 is the question point that is the nearest neighbor to the point q^* (i.e., the minimum value of the function $f(P_1)$) and located in the region W. To find the good question q_m for substitution efficiently, we perform the Best First Search (BFS) [15] with the R-Tree. BFS recursively visits the nearest question whose region is close to q^*. For efficiency, BFS uses a memory-resident heap $\mathcal{H}$ to manage all the questions in the R-tree that have been accessed. This continues until a question de-heaped from $\mathcal{H}$ is located in W. We note that because there may be more than one good question found as mentioned above, the actual best question should has the maximum discrimination degree among these questions such that the average discrimination degree of the generated test paper is maximized. Algorithm 1 presents the overall Tabu Search algorithm for the assessment constraint optimization.

5 Performance Evaluation

As there is no benchmark datasets available in the research community, we generate 4 large-sized synthetic datasets, namely D_1, D_2, D_3 and D_4 with number of questions of 20000, 30000, 40000 and 50000 respectively for performance evaluation. The values of each attribute in the 4 datasets are generated according to a normal distribution. Table 2 shows the summary of the 4 datasets. In addition, we have designed 12 test specifications with different parameters. The experiments are conducted in the Windows XP environment running on an Intel Core 2 Quad 2.66 GHz CPU with 3.37 GB memory. We evaluate the performance based on the 12 test specifications for each of the following 6 algorithms: GA, PSO, DE, ACO, TS and DAC-TS. We measure the runtime and quality of the generated test papers for each experiment. The 3 parameters of the DAC-TS are set experimentally as follows: $t_{TS} = 30$, $l_{TL} = 200$, $r = 0.6$.

To evaluate the quality of k generated test papers on a dataset $\mathcal{D}$ w.r.t. any arbitrary test paper specification $\mathcal{S}$, we use Mean Discrimination Degree and Mean Constraint Violation. Let $P_1, P_2, ..., P_k$ be the generated test papers on a question dataset $\mathcal{D}$ w.r.t.

Table 2. Test Datasets

	D_1	D_2	D_3	D_4
Number of Questions	20000	30000	40000	50000
Number of Topics	40	50	55	60
Number of Question Types	3	3	3	3

Algorithm 1 . Tabu Search for Assessment Constraint Satisfaction

Input:

$\mathcal{S} = (N, T, D, C, Y)$ - test paper specification; $P_0 = \{q_1, q_2, .., q_N\}$ - initial test paper; t_{TS} - short-term memory tenure; l_{TL} - long-term memory length; r - relaxation ratio; $\mathcal{R}$ - R-Tree index

Output:

P^* - Improved test paper

Process:

$\mathcal{P} \leftarrow \{P_0\}$; $MFT \leftarrow \emptyset$; $TS \leftarrow \emptyset$; $TL \leftarrow \emptyset$; $nbmax = 3l_{TL}$; $nbiter = bestiter = 0$
while P_{best} is not satisfied **and** $(nbiter - bestiter) < nbmax$ **do**
  $nbiter := nbiter + 1$; $optiter := optiter + 1$;
  for each q_i **in** P_0 **do**
    Compute 2-dimensional range W /* pruning search space*/
    $q_m \leftarrow$ **Best_First_Search**$(q_i, W, \mathcal{R})$;
    $P_1 \leftarrow \{P_0 - \{q_i\}\} \cup \{q_m\}$
    if $(q_i \notin TS$ **and** $(c_i, y_i) \notin TL)$ **or** $f(P_1) < f(P_{best})$ **then**
      Inserting new test paper P_1 into $\mathcal{P}$
    end if
  end for
  $P^* \leftarrow \underset{P' \in \mathcal{P}}{\text{argmin}}\ f(P_1)$; $\mathcal{P} \leftarrow \{P^*\}$ /* best move*/
  Update $MFT(c_m, y_m)$, Update $TS(q_i)$, Update $TL(c_m, y_m)$;
  if $f(P^*) < f(P_{best})$ **then**
    $P_{best} = P^*$; $bestiter = nbiter$; $optiter := 0$
  else if $optiter > 2l_{TL}$ **or** $\frac{f(P) - f_{best}}{f_{best}} > r$ **then**
    $optiter := 0$; $TS \leftarrow \emptyset$; $TL \leftarrow \emptyset$ /* memory relaxation*/
  end if
end while
return P^*

different test paper specifications $\mathcal{S}_i$, $i = 1..k$. Let E_{P_i} be the average discrimination degree of P_i. The *Mean Discrimination Degree* $\mathcal{M}_d^{\mathcal{D}}$ is defined as:

$$\mathcal{M}_d^{\mathcal{D}} = \frac{\sum_{i=1}^{k} E_{P_i}}{k}$$

The Mean Constraint Violation consists of two components: Assessment Constraint Violation and Content Constraint Violation. In Content Constraint Violation, Kullback-Leibler (KL) Divergence [11] is used to measure the topic distribution violation $\triangle C(\mathcal{S}_P, \mathcal{S})$ and question type distribution violation $\triangle Y(\mathcal{S}_P, \mathcal{S})$ between the generated test paper specification $\mathcal{S}_P$ and the test paper specification $\mathcal{S}$ as follows:

$$\triangle C(\mathcal{S}_P, \mathcal{S}) = D_{KL}(pc_p || pc) = \sum_{i=1}^{M} pc_p(i) \log \frac{pc_p(i)}{pc(i)}$$
$$\triangle Y(\mathcal{S}_P, \mathcal{S}) = D_{KL}(py_p || py) = \sum_{j=1}^{K} py_p(j) \log \frac{py_p(j)}{py(j)}$$

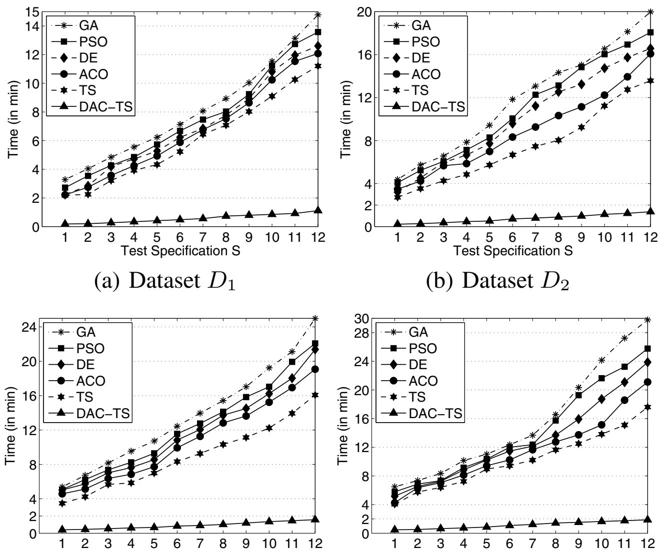
The Constraint Violation (CV) of a generated test paper P w.r.t. $\mathcal{S}$ is defined as:

$$CV(P, \mathcal{S}) = \frac{\lambda * \triangle T + \lambda * \triangle D + \log \triangle C + \log \triangle Y}{4}$$

where $\lambda = 100$ is a constant used to scale the value to a range between 0-100. The *Mean Constraint Violation* $\mathcal{M}_c^{\mathcal{D}}$ of k generated test papers $P_1, ..., P_k$ on a question dataset $\mathcal{D}$ w.r.t different test paper specifications $\mathcal{S}_i$, $i = 1..k$, is defined as:

$$\mathcal{M}_c^{\mathcal{D}} = \frac{\sum_{i=1}^{k} CV(P_i, \mathcal{S}_i)}{k}$$

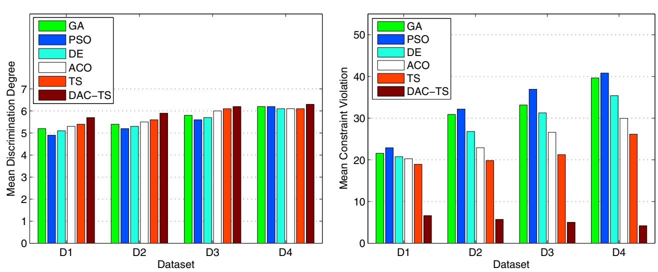
Figure 3 gives the runtime performance of the proposed approach in comparison with other techniques on the 4 datasets. The results have shown that DAC-TS outperforms other techniques in runtime. In Figure 3, it also shows that DAC-TS satisfies the runtime requirement as it generally requires less than 2 minutes to complete the paper generation process for various dataset sizes. In addition, the DAC-TS approach is scalable in runtime. Figure 4 shows the quality performance of DAC-TS and other techniques based on Mean Discrimination Degree $\mathcal{M}_d^{\mathcal{D}}$ and Mean Constraint Violation $\mathcal{M}_c^{\mathcal{D}}$ for the



(a) Dataset D_1 (b) Dataset D_2

(c) Dataset D_3 (d) Dataset D_4

Fig. 3. Performance Results Based on Runtime



(a) Mean Discrimination Degree (b) Mean Constraint Violation

Fig. 4. Performance Results based on Quality

4 datasets. As can be seen, DAC-TS has consistently outperformed other techniques. As such, DAC-TS is able to generate higher quality test papers than other techniques.

6 Conclusion

In this paper, we have proposed an efficient constraint-based Divide-And-Conquer Tabu Search (DAC-TS) approach for online test paper generation. The performance results have shown that the DAC-TS approach has not only achieved good quality test papers, but also satisfied the online runtime requirement even for large datasets in comparison with other techniques. Thus, the proposed research is particularly useful for Web-based testing and intelligent tutoring in an educational environment. For future work, we would like to combine the DAC-TS with the integer linear programming to further enhance the constraint satisfaction and runtime efficiency of the DAC-TS approach.

References

1. Beckmann, N., Kriegel, H.P., Schneider, R., Seeger, B.: The r*-tree: an efficient and robust access method for points and rectangles. ACM SIGMOD Record 19(2), 322–331 (1990)
2. Conejo, R., Guzmn, E., Milln, E., Trella, M., Prez-De-La-Cruz, J.L., Ros, A.: Siette: a web-based tool for adaptive testing. International Journal of Artificial Intelligence in Education 14(1), 29–61 (2004)
3. Glover, F., Laguna, F.: Tabu Search. Kluwer Academic Publishers (1997)
4. Gonzalez, T.F.: Handbook of Approximation Algorithms and Metaheuristics. Chapman & Hall/Crc Computer & Information Science Series (2007)
5. Guzman, E., Conejo, R.: Improving student performance using self-assessment tests. IEEE Intelligent Systems 22(4), 46–52 (2007)
6. Ho, T.F., Yin, P.Y., Hwang, G.J., Shyu, S.J., Yean, Y.N.: Multi-objective parallel test-sheet composition using enhanced particle swarm optimization. Journal of ETS 12(4), 193–206 (2008)
7. Hu, X.M., Zhang, J., Chung, H.S.H., Liu, O., Xiao, J.: An intelligent testing system embedded with an ant-colony-optimization-based test composition method. IEEE Transactions on Systems, Man, and Cybernetics 39(6), 659–669 (2009)
8. Hwang, G.J.: A test-sheet-generating algorithm for multiple assessment requirements. IEEE Transactions on Education 46(3), 329–337 (2003)
9. Hwang, G.J., Lin, B., Tseng, H.H., Lin, T.L.: On the development of a computer-assisted testing system with genetic test sheet-generating approach. IEEE Transactions on Systems, Man, and Cybernetics 35(4), 590–594 (2005)
10. Hwang, G.J., Yin, P.Y., Yeh, S.H.: A tabu search approach to generating test sheets for multiple assessment criteria. IEEE Transactions on Education 49(1), 88–97 (2006)
11. Kullback, S.: Information theory and statistics. Dover Publisher (1997)
12. Lee, C.-L., Huang, C.-H., Lin, C.-J.: Test-Sheet Composition Using Immune Algorithm for E-Learning Application. In: Okuno, H.G., Ali, M. (eds.) IEA/AIE 2007. LNCS (LNAI), vol. 4570, pp. 823–833. Springer, Heidelberg (2007)
13. Li, Q.: Guest editors' introduction: Emerging internet technologies for e-learning. IEEE Internet Computing 13(4), 11–17 (2009)
14. Manolopoulos, Y., Nanopoulos, A.: R-trees: Theory and Applications. Springer, Heidelberg (2006)
15. Roussopoulos, N., Kelley, S., Vincent, F.: Nearest neighbor queries. In: Proceedings of the ACM SIGMOD, pp. 71–79 (1995)
16. Rui, W.F., Hong, W.W., Ke, P.Q., Chao, Z.F., Liang, J.J.: A novel online test-sheet composition approach for web-based testing. In: Symposium on IT in Medicine & Education, pp. 700–705 (2009)