March 2004

Automated Simultaneous Assembly of Multi-Stage Testing for the Uniform CPA Examination[1]

Krista Breithaupt, American Institute of CPAs
Adelaide Ariel and Bernard Veldkamp, University of Twente

Paper Presentation at the Annual Meeting of the National Council on Measurement in Education,
San Diego, CA
April 2004

Cite as:

Breithaupt, K. Ariel, A., & Veldkamp, B. (2004). Balancing item exposure and optimality in automated assembly for multistage testing. Paper presented at the annual meeting of the National Council on Measurement in Education, San Diego CA.

[1] The authors gratefully acknowledge Wim van der Linden, Ron Hambleton and Richard M. Luecht. Their guidance and research were essential in evaluating and defining the MST model, the item bank design and the specification for automated test assembly for the computerization of the CPA examination. Any mistakes or oversights in this report are entirely the authors'.

Abstract

Some solutions used in the assembly of the computerized Uniform Certified Public Accountancy (CPA) licensing examination are offered as practical alternatives for operational programs producing large numbers of forms. The Uniform CPA examination will be offered as an adaptive multi-stage test (MST) beginning in April of 2004. Examples of automated assembly using mixed integer programming solutions in Optimization Programming Language software (OPL Studio 3.6.1© ILOG Inc.) illustrate MST design features and implementation strategies that can be generalized to other automated assembly problems.

A compromise between the best possible combination of test content, and sustainability and security over time is afforded by linear programming techniques that make use of mixed integer optimization algorithms. This method of formulating relative optimization functions to ensure a variety of constraints are always met for testlets of differing difficulties is also analyzed to evaluate the exposure of testlets and panels that is associated with the MST design. Technical information is also shared to assist other practitioners and researchers to emphasize feasibility and efficiency in their test construction problems. Some practical consequences from the selection of statistical targets on testlet exposure are illustrated. These results are described with respect to the fundamental principles of automated continuous test assembly and administration based on MST where testlet or item exposure projections and inventory rotation are critical concerns.

Introduction

The Uniform CPA licensing examination is a requirement for practice as a certified public accountant in the USA. The paper-based examination has been administered twice a year for over 70 years, and is taken by close to 400,000 candidates annually. There are benefits to be derived from the computerization of such high stakes examinations, especially one that is delivered to a large volume of candidates.

Computerized delivery makes it possible to offer continuous test administration, at the convenience of the candidate. Greater security is associated with the semi-adaptive administration model selected for the CPA examination. Computerization of assembly and delivery can also make randomization and strategic inventory planning easy to control. However, this transition to continuous computer-based administration requires a large item bank, and customized systems to assemble a large number of high-quality test materials on an ongoing basis.

The goal of this report is to share the design principles, and some details of implementation and quality assurance controls that were used to create the computerized version of the CPA examination. Our intention is to provide an illustrated example of some practical solutions to automated assembly problems where item pools may be limited and security of content is essential.

Multi-Stage Testing (MST) Design

The computerized version of the Uniform CPA examination will be delivered beginning in April of 2004. To qualify for licensure, each candidate passes four different examination sections. The administration model of the computer-based test (CBT) was formulated as a sequential (adaptive) testlet model, a variation of the design proposed by Luecht & Nungester, (1998). This CBT design was selected for improved precision in total scores, more efficient use of the item pool, and the advantage of tailoring the test experience for examinees of different ability levels.

The MST design used for the CPA Exam includes panels of five sets (testlets) of multiple-choice questions (MCQs) of equal length for each section. There are also complex performance simulations to measure accountancy skills that are administered after the MCQ portion of the examination is completed, described in detail elsewhere (DeVore, 2002). The five testlets and the simulation pair represent a panel. At any given time, up to forty panels may be available for administration. Figure 1 depicts only the MCQ testlets in one panel. Each MCQ testlet is targeted

at a specific difficulty level. At administration time, each candidate is given a random selection of eligible panels from the available test package in the test center. The first testlet the candidate receives is of average difficulty. Based on the performance of the candidate, the test driver will administer either a moderate or a difficult testlet in the second stage. At the completion of the second testlet, the candidates' performance on the first two testlets is evaluated to determine whether to administer a moderate or a difficult testlet at the third stage. Figure 1 is simplified for illustrative purposes. It is important to note that the CPA examination uses testlets that are either moderate (M) or difficult (D), and that these testlets overlap to a greater extent than is shown in the figure.

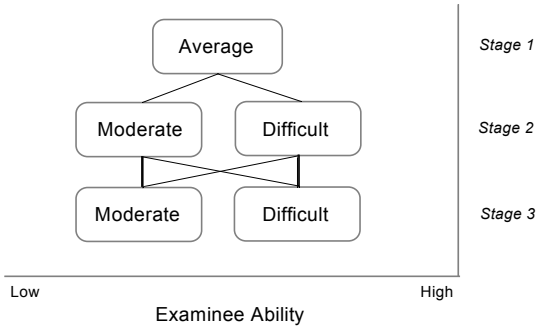

Figure 1: A three-stage adaptive multi-stage test design

To support continuous testing and to minimize content exposure, more panels have to be available. Direct implication to this policy is more testlets are needed. Of course, these testlets should be *parallel* to ensure uniformity and to balance the usage of content across administrations (Luecht, Brumfield, and Breithaupt, 2002). Parallel in this case can be defined by the IRT properties of the test questions and by content balancing as expressed in the test blueprint. An important assumption is that it is possible to produce large numbers of parallel testlets for each panel that meet the same testlet rules.

In the case of MST, the testlets are assembled in advance of administration in any test center. This enables the assembly design to be refined so that each completed set of testlets, when they are assembled into a panel, will exactly conform to content and other design constraints. However, it is important to emphasize that it is the rules (constraints and statistical targets) and basic units of construction (e.g. the multiple choice items and their codes) that are evaluated and

improved as the automated model is finalized. Subject-matter experts worked with our psychometric team over a period of about two years to review testlets and explicate their assembly requirements. At the time of the first operational build, there were no outstanding refinements to the automated process. Quality controls in the content development and assembly processes lead to consistently parallel testlets.

Testlet Assembly Method

In any continuous testing program where multiple tests are delivered, there exists a challenge to build tests that will each exactly meet all constraints imposed by the test blueprint. Typical test requirements focus on content coverage dictated by a publicized test blueprint and include a variety of other construction rules. A common rule imposes limitations on item enemies; certain items are not allowed to appear on the same test if they cue other items or are close variations of other items on that test form. There may also be rules for forced inclusion of items on key topics, or limits on the number of items related to similar content. It is important to carefully design the automated test assembly process to ensure that high quality tests are produced on a continual basis without over-use of pool content. The following paragraphs describe some design issues to consider in identification of the rules for testlet construction, and a method that may be used to solve the simultaneous assembly problem.

Design Issues

Four important decisions were required to finalize the testlet assembly design. These were solved in the following order:

1. Definition of the objective in assembly.
2. Location and kind of targets for IRT information.
3. Number of possible testlets.
4. Stringency of content constraints.

1. *Definition of the objective in assembly*

The mandated goal of our testing program is to protect the public by admitting only qualified CPAs to licensure. The jurisdictions use the examination result in consideration of whether the CPA has met all requirements for professional practice. Therefore, it is important to obtain optimal measurement precision in the area of the passing score. A secondary goal is to provide informative feedback to candidates who are not successful so they might prepare to re-take the examination. Therefore, a lesser goal was defined to obtain good measurement precision and coverage of a broad range of content just below the area of the passing score. Finally, a pervasive goal for all areas of any high stakes testing program is the security of test content to

preserve the validity of classification decisions that depend on test scores. In the assembly process, this security requirement led us to seek a method to make the best use of the item inventory.

2. Location and kind of targets for IRT information

The goals described above led to a decision on the second outstanding design issue. Test information was maximized within the area of the passing score for the difficult (D) testlets. The moderate (M) testlets were set to have most precision about one standard deviation below this value (expressed on the IRT ability scale, as theta). In order to make good use of the item inventory, the median difficulty of the item pool was used as an approximate midpoint between the M and D testlet targets. Also, a strategy of defining the M and D targets as relative was selected so that the best possible solution could be obtained based on the properties of the entire master item bank.

3. *Number of possible testlets*

Once the location and kind of targets for testlet information had been defined, and the content and other constraints had been determined, it was possible to begin trials against the master pool of items to determine how many testlets were possible at any particular time. Several factors must be considered, including size of the pool, dependencies among items (e.g. the size of item enemy sets, or interaction between item difficulty and content) and the minimum number of items required for adequate content coverage. Based on these factors, a rough estimate of the upper limit on the number of testlets can be conjectured.

Using the estimate for a maximum number of testlets, it was possible to determine the robustness of the master pool. A plan was implemented, whereby only a sub-set of this number of testlets would be used in any operational build. This was accomplished by returning a portion of the testlets to the master pool. In successive production cycles for later administration, testlets were assembled using only a sub-pool from master pool (including the unused reserved content and a selection of content used in prior administrations). In this way, it was possible to ensure that testlets of equivalent quality could be constructed for later administration. In addition, the empirical exposure of each item is used to filter subsequent sub-pools. These factors allow pro-active control of exposure at the earliest stage of assembly.

4. Stringency of content constraints

The published test blueprint defines what proportion of the examination will cover each content area. These are expressed as ranges, to allow some variation in coverage according to the available item pool. Competing goals exist in selecting the range of content to express in the

assembly constraints. It is desirable in simultaneous assembly to make the ranges as small as possible so that testlets will be exactly parallel (e.g. each would have the same number of questions on a particular content area). However, tightly constrained solutions will have the effect of limiting the number of possible testlets. So, ranges were set with an understanding of the exact contents of the item bank, within the allowable variation in the published test blueprint.

Solving the Testlet Assembly Problem

A general procedure for automated assembly of parallel tests requires items to be selected from an IRT-calibrated item bank to fill a desired shape of test information. A full treatment of modern test theory and item response theory (IRT) is outside the scope of this article. However, the reader is referred to Hambleton & Jones (1993) for a clear introduction of this topic. It might be sufficient to summarize that an IRT solution to the test assembly and scoring process was chosen because of benefits from pre-equating tests that will contain different content. Classical test scoring and assembly require a post-administration adjustment of scores to ensure the passing score on different forms will be comparable. This post-equating step is not required for IRT-based methods because the test content is pre-tested and items are calibrated prior to operational use. The IRT statistical properties of items (item parameter estimates) are known in advance and used in assembly and scoring to ensure equivalence of the passing score for candidates who respond to different content. The IRT parameters are the basis for determining the precision of scores (test information) along the ability scale. Test information can be plotted as a function of these parameters, and this test information function (TIF) is used to evaluate the equivalence of test scores from different forms.

An extension of the general assembly method using TIFs has reformulated the problem as one of linear optimization (Theunissen, 1985; van der Linden, in press; Veldkamp, 2001). A typical expression of the optimization problem for test assembly will maximize test information over a certain range of the IRT ability scale. Decision variables in the model will be binary to denote whether an item is selected or not. A mixed integer programming (MIP) solution for this kind of problem is available from industries where supply-chain management, scheduling and inventory planning depend on linear dependencies between entities with defined characteristics.

MIP solutions can be identified using a very large number of constraints and variables (such as we have in our testlet assembly problem). However, this kind of problem is notoriously difficult to solve. In our application of MIP, a large number of parallel testlets have to be simultaneously assembled to ensure equivalence in psychometric quality across forms, and to reduce the risk of over-use of our best content (e.g. highly discriminating items). Each additional rule or requirement

in the assembly model has a multiplicative effect on the total number of constraints that must be met.

A solution that meets all constraints is often computationally demanding, or even impossible. Van der Linden & Adema (1998) have proposed the use of a 'dummy test approach' as one efficient technique that will reduce computational difficulty in assembling parallel tests. Other proposed techniques have proposed heuristic methods (e.g. Luecht & Burgin, 2003) to decrease the time required to reach a reasonable solution. However, these methods may depend on relaxed constraints related to content, and may not provide optimal solutions.

A general model for the assembly process defined by van der Linden and Boekkooi-Timminga (1989) was adapted for use in our operational production of MST testlets. Boekkooi-Timminga defined the assembly model by maximizing the lower bound of the testlet information at the specific target ability point (also termed a '*maximin'* approach). This lower bound value represents a relative target for the TIF and is represented as *y* in the model defined as follows:

Maximize *y,*

subject to:

(1) Relative targets for any testlet *t* for every θ_k value, using weight w_t (w_t > 0)

$$\sum_{i=1}^{I} x_{it} * I_i(\theta_k) \geq w_t * y, \quad t = 1, \ldots, T$$

(2) Fixed number of items for all testlets

$$\sum_{i=1}^{I} x_{it} = n, \quad t = 1, \ldots, T$$

(3) Lower (n_t) and upper ($n^{(t)}$) content bounds for items of each type, V_t

$$n_t \leq \sum_{i \in V_t} x_{it} \leq n^{(t)}, \quad t = 1, \ldots, T$$

(4) Critical content, C_t , must be included

$$\sum_{i \in C_t}^{I} x_{it} \geq 1, \quad t = 1, \ldots, T$$

(5) Limitation on the number of items from each enemy item set, *S*

$$\sum_{i \in S} x_{it} \leq n^{(s)}, \qquad t = 1, \ldots, T$$

(6) No items are shared on any testlets

$$\sum_{t=1}^{T} x_{it} \leq 1, \qquad i = 1, \ldots, I$$

(7) A binary decision indicates whether an item is assigned to a particular testlet

$$x_{it} \in \{0,1\}$$

Expression 1 defines the lower bound on testlet information that will be maximized in the simultaneous build of a set of testlets. This expression was expanded in our case to include more than one theta value to be weighted for each M and D testlet. For example, there were three theta values needed to represent M testlets, and the first constraint included subordinate expressions for each theta value. When a common theta value is expressed as a target for M and D testlets, the testlet with a higher weight at that target will be favoured for the selection of discriminating items.

Perhaps the most distinctive element in this model should be emphasized. It is essential that the model maximizes the minimum information (or weighted information) for all testlets in a single solution. This ability of the MIP allows us to create the M and D testlets *simultaneously* and to ensure the best possible combination of items into testlets has been found as our solution with respect to all the constraints in the assembly problem. Each expression of a constraint is exactly met by the final optimal solution to the assembly problem. This computationally intensive process can be completed using available software in a reasonable amount of time. The testlet assembly using OPL Studio for any one section of our examination applied approximately 50,000 constraints and 60,000 variables. Using a common desktop computer (1.2 GHz), the solution time required varied from 30 minutes to just under an hour to create all the required testlets in any one section.

The optimization solution may also be tailored during successive trial runs to conform to the needs of an individual testing program or item pool. It is possible to adjust the relative target (in this case testlet information) of the M and D testlets, respectively, by using the weight value in the first constraint. This flexibility is useful in making good use of the item sub-pool, and meeting our design decision to offer precision in the area of the cut-score. It is also possible to control the

degree of overlap between the M and D testlets by placing specific weights on their respective tails. For instance, D testlets might need items of difficulty level that overlap with the items needed by M testlets. When a higher weight is given to the D testlet, those items will tend to be selected. It is equally possible to place weights at three points on three M testlet targets to ensure even use of the item bank across the ability scale. As a consequence we can ensure useful feedback for candidates just below the passing score.

When automated assembly programs are not able to create all required testlets, the MIP problem is '*infeasible'*. Detecting the causes of infeasibility can be complex when the assembly problem has many model constraints. An overview of literature on infeasibility analysis is given in Huizing, Veldkamp, and Verschoor (2003). Some researchers have offered techniques to detect infeasibility (e.g. Huitzing, in press), whereas others offer an algorithm to flexibly deal with the constraints while choosing items for a test (e.g. Stocking and Swanson, 1993). When this flexibility or relaxation of constraints is not desirable (e.g. constraints are absolute to ensure equality across testlets or administrations) alternatives must be considered.

One approach is to redesign the master pool (van der Linden, Veldkamp, and Reese, 2000). Content developers working on operational testing programs often informally follow this long-term strategy. However, advanced techniques now exist to model and direct inventory development using a MIP formulation for the problem. A simple strategy that might be useful in some testing programs is to *duplicate* some items, or allow items to be shared across testlets (Luecht, Brumfield, and Breithaupt, 2002).

Infeasibility was only encountered during the trial assembly runs when the quality of the master pool was less well understood. There were no infeasibility problems encountered in the assembly solutions used for our operational production build. All the resulting testlets were equivalent in their absolute adherence to the content, statistical and other specifications.

An example of the overlapping testlet information functions (TIFs) is presented in Figure 2. Each curve represents the information from a single testlet across the theta scale. The TIFs are very close in the amount of information provided at equivalent theta points. They are plotted at only 13 points on the ability scale, and would otherwise be presented as smooth curves. TIFs at the left (lower) ability level are the M testlets and have a lower information target and a smaller weight, compared with the D testlets on the right. The scripting ability in the OPL Studio software allows the resulting testlets to be transmitted as output in a variety of forms. The plotted testlets are automatically generated as OPL Studio writes directly into Microsoft Excel © Spreadsheets.

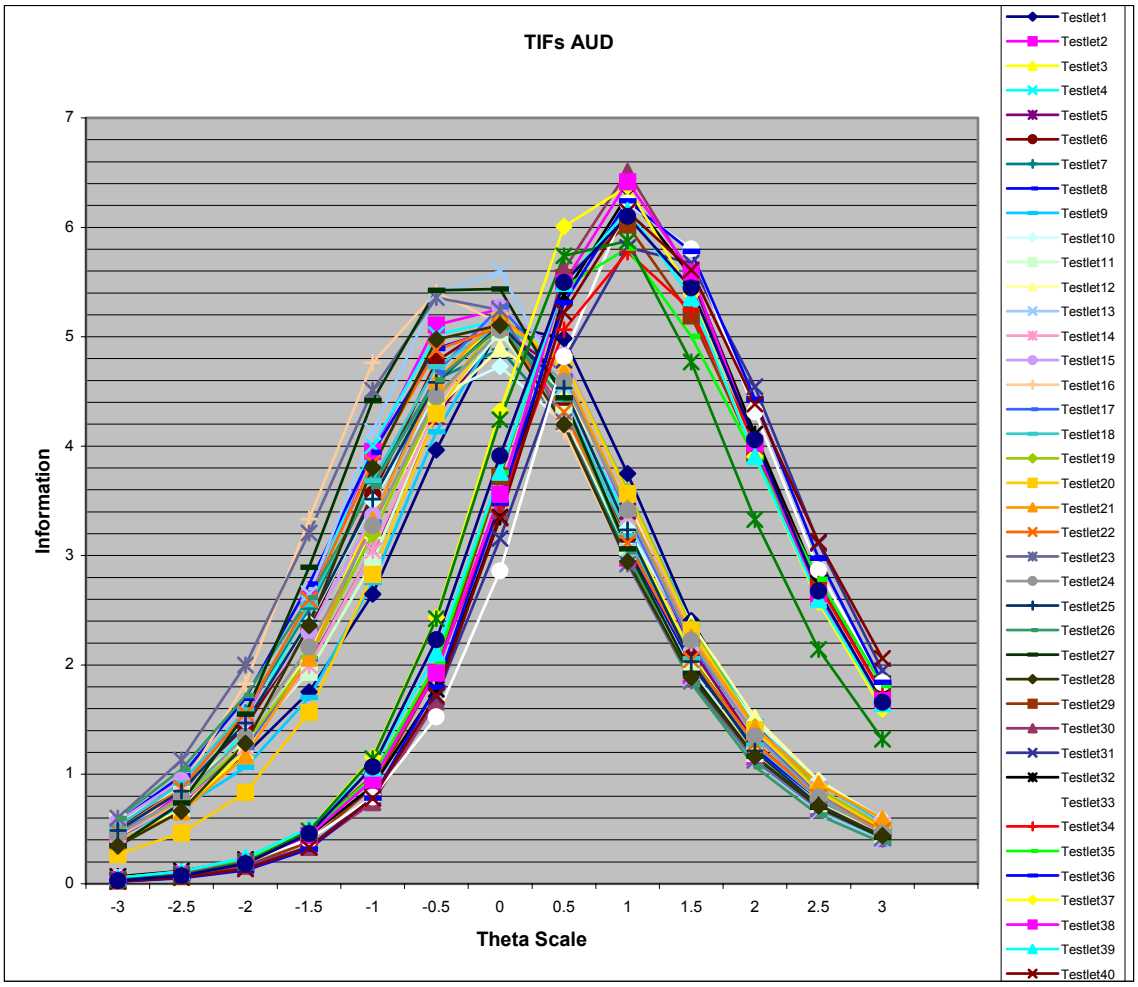

Figure 2: Test information functions from simultaneous testlet assembly

The script files and optimization models also deal with the problem of assembly and assignment of pre-test questions in each panel. The linear programming software simultaneously solves the problem representing MCQs to be scored, and MCQs that will be placed in the examination to gather pre-test results for bank development. Pre-test blocks are designed to fill weak areas in the item bank, and are assigned to each stage of the MST. It may be important to note that pre-test questions are assigned to both the moderate and difficult testlet at any stage to reduce calibration problems related to constrained samples.

When the solution run has been completed, the psychometric and content teams review results using a set of informative text files, also generated by the script. Output files that describe key properties of the testlets were designed created for this review, and include such information

content coverage and enemy conditions. In addition to quality control files, a solution input file is generated by the script file to convey the composition of testlets to the next step in panel assembly.

Panel Construction

The panel construction step is accomplished via a custom software tool, but could easily be completed using logic implemented by linear programming rules similar to those used to assemble the testlets. The panel builder tool reads the testlet build output and places together five testlets on a single panel for the MST design, according to specific rules. These rules include limits to enemy items across testlets, placement of items for pre-testing, content balancing across routes and a maximum re-use rule for any testlet on panels, among others. Panel builder also generates the routing score table that is used by the test driver in the field. The panel builder execution requires a few minutes to select rule conditions, and is completed in a matter of seconds.

Creation of the routing rules is an important activity of the panel assembly software. The panel builder uses IRT properties of the testlets to identify a cut-off score on an estimated number correct scale that directs the test driver to administer the next testlet. Specifically, the point of intersection of each pair of M and D testlets is used as the pivot or cut-score point. There is a cut-score for the stage 2 decision, and for the stage 3 decision. This routing rule is enforced by the test administration software in the test center, based on the number correct score the candidate obtained (a specific routing decision is enforced for each possible combination of testlets a candidate may have answered). The routing table that is packaged for the test driver for each panel includes these number-correct cut-points.

Until this step in the process, all quality control work is completed using the features of items, testlets and panels that are represented as data input to the automated test assembly process. The actual content of the test questions (stems, exhibits and distracters for MCQs) is not needed during the build process. This procedure affords more control over the test content, and reduces security risks to the production process. The result from panel building is like a recipe for each examination, including all required instructions and none of the ingredients. All features of the test content important for valid panel and testlet assembly are extracted as codes in the item bank. The success of the build depends on accurate and consistent coding of the item bank.

Conclusions

The design of the computer-based Uniform CPA examination is one example of a solution for the complex test assembly problem. Many advantages for operational testing programs exist when multiple competing and complex test design rules can be automated. It is possible to make use of existing software, such as OPL Studio® to solve the mathematical expression of this problem as while retaining desirable psychometric properties for testlets or test forms. Many of these principles and practical examples can be generalized and could be applied by other agencies or academic programs concerned with continuous test production for computerized delivery.

As we continue to research and modify these systems and procedures, the AICPA Examination team hopes to extend this test assembly approach to solve existing production problems and to reduce our reliance on resources, in particular, the time required from subject-matter experts. The attention of our review committees is already shifting from the traditional evaluation of the completed examination forms, to quality assurance on the content development and rules for automated assembly. Future improvements have been planned, including a simplified interface for the testlet assembly system that would not require detailed understanding of the underlying mathematical expressions, and optimization models to produce testlets and panels for adjacent testing periods simultaneously. These developments are intended to have cost reduction benefits, to improve our inventory controls and to increase the security of our test content. This illustration is one instance where advances in psychometric theory and practice lead to benefits beyond elegant mathematics, and offer immediate and practical gains for high-stakes testing programs.

References:

DeVore, R. (2002). Considerations in the Development of Accounting Simulations. Paper presentation at the annual meeting of the National Council on Measurement in Education, New Orleans, LA. Also as a Technical Report available form AICPA.

Hambleton, R.K. & Jones, R.W. (1993). Comparison of classical test theory and item response theory and their applications to test development. Educational Measurement: Issues and Practice. Instructional Topics in Educational Measurement Series (Fall) 38-47.

Huitzing H.A. (in press). Using Set Covering with Item Sampling to analyze infeasibility of linear programming test assembly models. Accepted for publication in *Applied Psychological Measurement.*

Huitzing, H.A., Veldkamp, B.P. & Verschoor, A.J. (2003). Infeasibility in automatic test assembly models: a comparison study of different methods.

Luecht, R. M. & Nungester, R (1998). Some practical examples of computer-adaptive sequential testing. Journal of Educational Measurement, 35 (3) 229-249.

Luecht, R. M. (1998). Computer-assisted test assembly using optimization heuristics. Applied Psychological Measurement, 22 224-236.

Luecht, R.M., Brumfield, T. & Breithaupt, K. (2002). *A testlet-assembly design for the uniform CPA examination*. Paper presented at the Annual Meeting of the National Council on Measurement in Education, New Orleans, LA.

Luecht, R.M, & Burgin, W. (2003). Test Information Targeting Strategies for Adaptive Multistage Testing Designs. Paper presentation at the annual meeting of the National Council on Measurement in Education, Chicago. IL.

Stocking, M.L., & Swanson, L. (1993). A method for severely constrained item selection in adaptive testing. *Applied Psychological Measurement, 17*, 277-292.

Theunissen, T.J.J.M. (1985). Binary programming and test design. Psychometrika, 50, 411-420.

Van der Linden, W.J. (2000). Constrained adaptive testing with shadow tests. In W.J. van der Linden & C.A.W. Glas (Eds.), Computerized adaptive testing: theory and practice (p 27-52). The Netherlands: Kluwer Academic Publishers.

Van der Linden, W.J. Linear Models for Optimal Test Design.(in press). Kluwer Pub: NY.

Van der Linden, W.J. & Adema, J.J. (1998). Simultaneous assembly of multiple test forms. Journal of Educational Measurement, 35 185-198.

van der Linden, W.J., & Boekkooi-Timminga, E. (1989). A maximin model for test design with practical constraints. Psychometrika, *54*, 237-247.

Veldkamp, B.P. (2001). Principles and methods of constrained test assembly. Thesis, Enschede, The Netherlands: University of Twente.