DOCUMENT RESUME

ED 473 526 TM 034 734

AUTHOR	Veldkamp, Bernard P.; van der Linden, Wim J.; Ariel, Adelaide
TITLE	Mathematical-Programming Approaches to Test Item Pool Design. Research Report.
INSTITUTION	Twente Univ., Enschede (Netherlands). Faculty of Educational Science and Technology.
REPORT NO	RR-02-09
PUB DATE	2002-00-00
NOTE	28p.
AVAILABLE FROM	Faculty of Educational Science and Technology, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands. E-mail: Fox@edte.utwente.nl.
PUB TYPE	Reports - Descriptive (141)
EDRS PRICE	EDRS Price MF01/PC02 Plus Postage.
DESCRIPTORS	*Adaptive Testing; *Computer Assisted Testing; *Item Banks; Test Construction; *Test Items

ABSTRACT

This paper presents an approach to item pool design that has the potential to improve on the quality of current item pools in educational and psychological testing and thus to increase both measurement precision and validity. The approach consists of the application of mathematical programming techniques to calculate optimal blueprints for item pools. These blueprints can be used to guide the item-writing process. Three different types of design problems are discussed: (1) item pools for linear tests; (2) item pools for computerized adaptive testing (CAT); and (3) systems of rotating pools for CAT. The paper concludes with an empirical example of the problem of designing a system or rotating item pools for CAT. (Contains 2 tables and 22 references.) (Author/SLD)

Mathematical-Programming Approaches to Test Item Pool Design

Research Report 02-09

Ariel Adelaide
Bernard P. Veldkamp

faculty of
EDUCATIONAL SCIENCE AND TECHNOLOGY

University of Twente

Department of
Educational Measurement and Data Analysis

Mathematical-Programming Approaches to Test Item Pool Design

Bernard P. Veldkamp

Wim J. van der Linden[1]

Adelaide Ariel

[1]This paper was written while the second author was a Fellow of the Center for Advanced Study in the Behavioral Sciences, Stanford, CA.

Abstract

This paper presents an approach to item pool design that has the potential to improve on the quality of current item pools in educational and psychological testing and hence to increase both measurement precision and validity. The approach consists of the application of mathematical programming techniques to calculate optimal blueprints for item pools. These blueprints can be used to guide the item-writing process. Three different types of design problems are discussed, namely for item pools for linear tests, item pools computerized adaptive testing (CAT), and systems of rotating item pools for CAT. The paper concludes with an empirical example of the problem of designing a system of rotating item pools for CAT.

Keywords: Item Pool, Item-Pool Design, Item Response Theory, Mathematical Programming, Optimal Test Assembly.

Introduction

In the early days of testing, tests typically had a linear format and once the test became obsolete, its items were just thrown away. In hindsight, this type of testing involved a waste of efforts and time. A more efficient type of testing is using item banking. In item banking, test items are written on a continuous based and tests are assembled to be optimal from pools of items in the item banking system. Items in the system are reused and after each administration the response data can be used to update the estimates of their statistical properties.

The introduction of item banking in testing has led to the introduction of techniques of automated test assembly. These techniques allow test assemblers to declare a set of specifications for the test they want from an item pool and to delegate the actual assembly process to a computer algorithm. An important class of algorithms is known by the name of Optimal Test Assembly (OTA). The majority of these techniques are based on the application of mathematical programming. Typically, these techniques require the formulation of a test assembly model with an objective function that maximizes the measurement precision of the tests and a set of constraints to guarantee that the test meets its specifications, for example, with respect to test length, content, or test format. A review of these techniques is given in van der Linden (1998).

The fact that tests are assembled to be optimal does not necessarily imply that their quality is perfect. Generally, a test can never have a better quality than permitted by the item pool from which it is assembled. Because the technique of designing good item pools is still in its infancy, item pools currently in use in educational and psychological testing are often unbalanced. For example, they may consist of a small set of high-quality items that are often selected for administration, while the majority of the items are seldom used.

The disadvantages of using such item pools became most obvious when computerized adaptive testing (CAT) was introduced. The first real-life CAT programs appeared to be vulnerable to item security problems because of organized efforts to memorize the subset of popular items in the pool. These efforts were successful because only a small set of items needed to be memorized to compromise the pool. Though item-exposure control methods can be applied to guarantee a maximum exposure rate for the items in the pool, such methods do not fix another

problem with unbalanced items pools, namely the waste of time and resources involved in writing and pretesting items that are seldom used.

The best way to overcome these problems is more systematic design and development of item pools. One of the avenues to improve item pool design is to begin with the calculation of an optimal *blueprint* for the item pool to guide the item-writing process. A blueprint is a document specifying what attributes the collection of items in the pool should have to serve its testing program in the best possible way. The current paper focuses on the problem of how to calculate an optimal blueprint for an item pool. Two different methods for calculating such blueprints will be presented, one method for item pools in a testing program with tests with a linear format and one for pools in a CAT program. In addition, we will present a method to design a system of rotating item pools for use in CAT.

The actual task of writing test items to a blueprint is difficult. The difficulty does not reside so much in the need to write items with predetermined content attributes of the items as well as to realize items with statistical attributes, such as p-values, item-test correlations, and IRT parameters, with predetermined values. It is common experience that the values of statistical attributes of *individual* items are only loosely predictable. At the same time, however, at the level of a pool of items, statistical attributes often show persistent patterns of correlation with content attributes. In this paper, these patterns are used to calculate the blueprint for a new item pool.

Overview of the Literature

Boekkooi-Timminga (1991) presented a method of item pool design for the assembly of linear test forms. The method is based on the technique of integer programming. It can be used to optimize the design of item pools that have to support the assembly of a future series of linear test forms. The method assumes an item pool calibrated under the one-parameter logistic (1PL) or Rasch model. The methods in the current paper are also based on mathematical-programming techniques. One of the differences with the method in Boekkooi-Timminga is that Boekkooi-Timminga's method follows a sequential approach calculating the numbers of items required in the pool for each individual test form at a time, where the methods in the current paper use a simultaneous approach.

A description of the process of developing item pools for CAT is given in Flaugher (1990) and Segall, Moreno, and Hetter (1997). Both authors outline several steps in the development of item pools. Flaugher discusses current practices at these steps. Segall, Moreno, and Hetter explain the procedure that is followed to construct item pools for the CAT version of the Armed Services Vocational Aptitude Battery (ASVAB). A common feature of the process described in Flaugher and the method in the present paper is the use of computer simulation in the design process. However, in Flaugher's outline, computer simulation is used to evaluate the performance of an item pool once the items have been written and field-tested whereas in the current paper computer simulation is used to design an optimal blueprint.

A rather different approach is described by both Stocking and Swanson (1998) and Way, Steffen and Anderson (1998; see also Way, 1998). They address the problem of designing a system of rotating item pools for CAT. This system assumes the presence of a master pool from which operational item pools are generated. A basic quantity in this method is the number of operational pools each item should be included in. By manipulating this number, desired exposure rates for the test items can be set. The same goal is realized by one of the methods in the current paper; however, this method is based on the application of an entirely different type of mathematical programming.

Analysis of Design Problem

Before focusing on the problem of designing an item pool, we have a closer look at the notion of test specifications. Test specifications can be categorized in several ways. In optimal test assembly, test specifications are formulated as constraints in a mathematical programming model. Based on the mathematical shape of these constraints, test specifications can be characterized as constraints on categorical item attributes, on quantitative item attributes, and on inter-item dependencies (van der Linden, 1998).

Categorical Constraints

Categorical item attributes partition the item pool into a series of subsets. Examples of such attributes are item content, cognitive level, format, author, and answer key. A test specification that is formulated as a categorical constraint generally constrains the

distribution of the items in the test over the subsets. If the items are coded by multiple attributes, their Cartesian product partitions the item pool. Test specifications can then also be formulated as a joint or conditional distribution over this partition.

A natural way to represent categorical item attributes is by a table. A fictitious example for a mathematics test is given in Table 1. One item attribute is content, C, with levels Geometry, Algebra, and Statistics); the other is item type, T (with levels Basic Skills and Application). In Table 1, the distribution of the items in the pool is

[Insert Table 1 about here]

represented by the numbers n_{ij}, $n_{i.}$, $n_{.j}$, and $n_{..}$ which are the numbers of items in cell (i,j), row i, column j, and the total table, respectively. If a test is to be assembled from this pool, a number of items have to be selected. Let the number of items to be selected from each subset be denoted as r_{ij}, $r_{i.}$, $r_{.j}$, and $r_{..}$. We may impose the following set of constraints on the selection of items in the test:

1. Number of Geometry items testing Application equal to eight ($r_{12} = 8$);
2. Number of Basic Skills items equal to nine ($r_{.1} = 9$);
3. Number of Geometry items equal to four ($r_{1.} = 4$);
4. Total number of items equal to 25 ($r_{..} = 25$).

Note that this set of constraints not only fixes certain numbers of items from some of the cells in Table 1 directly, but also constrains the numbers from the other cells in the table. For example, the first and last constraints together imply that the number of Application items on Algebra or Statistics is equal to four. Mathematically, this constraint can be denoted as $r_{12}+r_{13}=4$. This example shows that the same set of test specifications can be represented by different sets of mathematical constraints. From the theory of mathematical programming it is known that some of these sets are more efficient than others. It is not the focus of this paper to discuss the efficiency of constraint formulation; for a paper on this topic, see Veldkamp (submitted).

Quantitative Constraints

Unlike categorical constraints, quantitative constraints do not impose bounds on numbers of items. Instead, they impose bounds on a function of the values of the items on a quantitative attribute, mostly on their sum or average. Examples of quantitative item attributes are: word counts, exposure rates, values for item response theory (IRT) information functions, expected response times, and such classical item parameters as p-values and item-test correlations.

As an example of a quantitative constraint, consider a constraint on response time that imposes a bound on the sum of the expected response times of the items in the test for the examinees. This type of constraints is useful if examinees have a fixed amount of time available to take the test. It is important to note that though each possible combination of items in the test defines a unique sum of expected response times, the reverse does not hold. Unlike categorical attributes, constraints with quantitative attributes have no one-one correspondences with item distributions. Instead, they imply sets of different distributions each of which feasible with respect to the constraint. Later in this paper, this property is exploited to choose a distribution to represent a quantitative constraint that is *optimal* with respect to an objective function.

Observe that it is possible to discretize quantitative item attributes and represent them by a table, just as categorical attributes. For example, if we discretize the expected response times of the items, t_i, their scales of possible values is replaced by a finite grid of values, t_{id} , with d=1,...,D. The number of points on the grid as well as their spacing is free.

Below, we will use Q to represent the table defined by the product of joint grid for all quantitative attributes in the test assembly model, which will have an arbitrary cell denoted by q. Likewise, the symbol C is used to represent the table defined by all categorical attributes in the model, with an arbitrary cell denoted by $c \in C$. A cell in the joint table defined by C and Q will be denoted as $(c,q) \in C \times Q$. An optimal blueprint for an item pool is a table $C \times Q$ with optimal values of n_{ij} for its cells.

Constraints on Inter-Item Dependencies

The defining characteristic for this type of constraints is that it deals with relations of exclusion and/or inclusion among items in the pool. Two items exclude each other, for example, if one item contains a clue to the answer to the other item. We will call such items "enemies". In test assembly, it is possible to constrain the test to have no more than one item from each known set of enemies. Usually, the occurrence of enemy sets in an item pool is not planned; they just happen to be there. In practice, we deal with such sets by distributing the items in them over different test forms. Also, in practice the number of enemy sets in an item pool is generally low. The position taken in this paper is that the presence of enemies is a problem of *test assembly*--not of item pool design. It will therefore be ignored in the remainder of this paper.

However, some other types of constraints on interdependent items have to be included in the design process. Items can be organized around common stimuli, for example, a reading passage in a reading comprehension test or a description of an experiment in a biology test. We will use "item sets" as a generic term for this item format. Typically, the items in the sets in the test are selected from larger sets available in the pool. If so, constraints have to be added to the test assembly process to define how many items in the item set can be selected in the test. Besides, selecting item sets often involves constraints on categorical (e.g., content) and quantitative (e.g., word counts) attributes of the stimuli. Methods for designing item pools should be able to handle such complicated relationships.

Methods of Item Pool Design

Design methods have been developed for item pools for different types of tests. In this section, we will describe methods for designing pools for linear tests with item sets and for use in computerized adaptive testing (CAT). In addition, a method for designing a set of rotating item pools in CAT will be discussed.

Linear Tests

A design method for pools for linear tests with item sets can be based on the following three-stage procedure introduced in van der Linden, Veldkamp and Reese (1998):

1. A blueprint for a pool of *items* is designed using an integer-programming model ignoring the item set structure. The blueprint is calculated using a mathematical programming model that constrains the distributions of the items over their categorical and quantitative attributes. The model has an objective function that minimizes the costs of item writing.
2. A blueprint for a pool of *stimuli* for the item sets is designed using the same methodology as for the pool of items. The model now constrains the distribution of the stimuli over their categorical and quantitative attributes and the objective function minimizes a cost function for writing the stimuli.
3. Items and stimuli in the two blueprints are assigned to the each other to form a new blueprint for a pool of *item sets*. The assignment is done using a separate mathematical programming model that constrains the assignment to deal with the numbers of items available in the various cells of the CxQ table and the

numbers required in the item sets. The objective function is of the same type as above.

We will now discuss the mathematical programming models in somewhat more detail

Calculating Item Pool Blueprint. The objective function of the integer-programming model minimizes a cost function for writing the items. In general, costs of item writing are hard to assess. For some combinations of attributes it is easier to write items that for others. When no knowledge about the item writing process is available, a simple approach is to base the cost function on the distribution of the items in a typical previous pool.

Let x_{cq} denote the frequency of the items in cell (c,q) in a table for a previous pool. These frequencies contain information on the efforts involved in writing items for the various cells in the table. Cells with relatively large frequencies represent combinations of categorical and quantitative attribute values that tend to go together often; apparently, such items are easy to produce. On the other hand, empty cells seem to point at combinations of attribute values that are difficult to produce. A monotonically decreasing function of x_{cq}, denoted as $\varphi(x_{cq})$, will be used as an empirical measure of the efforts involved in writing items with the various possible combinations of attribute values. A simple cost function is $\varphi(x_{cq}) = x_{cq}^{-1}$, which requires $x_{cq}>0$. Other choices are possible tough.

We assume that the item pool has to serve the assembly of $f=1,\ldots,F$ test forms, each with possible different specifications. The integer-programming model for designing a blue print for a pool of items for these forms can be formulated as follows:

$$\text{Minimize} \sum_{f} \sum_{c} \sum_{q} \varphi_{cq}\, n_{fcq} \qquad \text{(minimizing costs)} \quad \textbf{(1)}$$

subject to

$$\sum_{c} \sum_{q} I_q(\theta_k)\, n_{fcq} \geq T_f(\theta_k),\ f=1,\ldots,F,\ k=1,\ldots,K, \qquad \text{(test information)} \quad \textbf{(2)}$$

$$\sum_{q} n_{fcq} \geq n_{fc},\ f=1,\ldots,F,\ c=1,\ldots,C, \qquad \text{(categorical constraints)} \quad \textbf{(3)}$$

$$\sum n_{fcq} \geq n_{f_q}, f=1,...,F, q=1,...,Q, \quad \text{(quantitative constraints)} \quad \textbf{(4)}$$

$$n_{fcq}=0,1,..., f=1,...,F, c \in C, q \in Q. \quad \text{(integer decision variables)} \quad \textbf{(5)}$$

The objective function in Equation 1 minimizes the sum of the item writing costs across all items in the F forms. For each form the constraints in Equation 2 require the test information function at θ_k, $k=1,...,K$, to be larger than a predetermined set of target values. The objective function in Equation 1 guarantees that these bounds are approached from above. The categorical constraints imposed on the forms are in Equation 3. Lower bounds n_{fc} are set only; the objective function in Equation 1 guarantees that the constraints are always satisfied as equality at optimality. The same holds for the quantitative constraints in Equation 4.

Solving the model in Equation 1-5 gives us the set of optimal numbers of items in the table, n_{fcq}, that together constitute the blueprint for the item pool.

Calculating Stimuli Pool Blueprint. The method for calculating a blueprint for a pool of stimuli is formally similar to the one for designing the items. Tables C' and Q' are now defined for the sets of categorical and quantitative attributes that describe the stimuli in the test forms the item pool has to support. It should be observed that table Q' is expected to be much smaller than Q, because psychometric attributes for stimuli are rare. Sometimes item sets have to meet constraints with respect to *aggregated* statistical attributes, such as sums of p-values or average b_i values. However, such aggregates belong to the set of items associated with a stimulus--not to the stimulus itself. Constraints on aggregated attributes are therefore dealt with in the item-assignment model below.

The integer-programming model for the blueprint of stimuli follows directly from Equations 1-5. The cost function $\varphi_{c'q'}$ can be derived from the distribution of stimuli in the previous item pool. If the number of stimuli is small and the number of stimulus attributes large, taking the inverse of the table frequencies may not work well. If so, other cost functions must be considered. Just as in the previous model, the bounds in the categorical and quantitative constraints represent the specifications for the item sets in the various test forms.

Assigning Items to Stimuli. The final step is to assign the items in the CxQ table to the stimuli in the $C'xQ'$ table. A regular assignment model from mathematical programming can be applied to optimize this step. Such a model has decision variables for each combination of cell in CxQ and $C'xQ'$ that take the value 1 if an item from the cell in CxQ is assigned to a stimulus in the cell in $C'xQ'$ and the value 0 otherwise. The model also has an objective function that is to be maximized or minimized. Like the previous two models, a function representing the costs of assigning an item to a stimulus is used. The cost function, $\varphi_{cqc'q'}$, is defined on the Cartesian product of the tables CxQ and $C'xQ'$. This function represents the costs of writing an item with attributes (c,q) for a stimulus with attributes (c',q'). The constraints in the assignment model are needed: (1) to represent the maximum and minimum number of items and stimuli available in each cell; (2) to constrain the size of the item sets that have to be formed; and (3) to deal with possible aggregated quantitative items attributes that item sets are required to have. For a detailed example of an assignment model, see van der Linden, Veldkamp & Reese (2000).

Computerized Adaptive Testing.

The process of designing on optimal blueprint for an item pool for use in CAT is somewhat more complicated. One of the reasons for this is that in CAT the goal is to have a test for each examinee with optimal information at his/her ability level. It is thus impossible to set a target for the test information function as in Equation 3. Another complication is that the number of different test forms is potentially as large as the number of examinees.

In Veldkamp and van der Linden (2000), an alternative method is described for designing a blueprint for CAT item pools. The method shares some of its logic with the method for developing blueprints for paper-and-pencil testing. Like the previous method, the set of specifications for the CAT is analyzed and all item attributes figuring in the specifications are identified. The result of this step is again a classification table that is the product of all categorical and quantitative item attributes. Each cell of the table represents a possible type of item in the pool.

However, unlike the design problem for linear tests, to obtain the optimal numbers n_{cq} for the table, we do not solve a mathematical programming model but run a computer simulation of the CAT for the ability distribution of the intended

population of examinees. In this simulation, the "items" available are the cells in the table, the examinees are sampled from an ability distribution estimated from historic data, and the test specifications are imposed on the CAT through a shadow test approach. (The notion of CAT with shadow tests will be explained below.). During the simulation, the number of times an item is selected from a cell in the table is counted. The blueprint for the item pool is calculated from these counts.

CAT with Shadow Tests. The shadow test approach has been developed to guarantee that all test specifications in a CAT are met, even when the items are selected adaptively (van der Linden & Reese, 1998, van der Linden, 2000). The items in the CAT are selected using a two-stage procedure. First, a linear test (=shadow test) of the required length of the CAT that meets all constraints is assembled from the pool. The test is also required to contain all items already administered and is assembled to be optimal at the examinee's current ability estimate. Second, the item to be administered is selected from the free items in the shadow that is optimal contribution at the ability estimate. As a result of this two-stage selection procedure, each adaptive test meets all constraints and has items that are always optimal.

The algorithm for constrained CAT with shadow tests can be summarized as follows:

1. Choose an initial value of the examinee's ability parameter θ_k.
2. Assemble the first shadow test such that all constraints are met and the objective function is optimized.
3. Administer an item from the shadow test with optimal properties at the current ability estimate.
4. Update the estimate of θ_k as well as all other parameters in the test assembly model.
5. Return all unused items to the pool.
6. Assemble a new shadow test fixing the items already administered.
7. Repeat Steps 3-5 until all n items have been administered.

Integer Programming Model in CAT Simulations. In the CAT simulations, an integer-programming model is used to assemble the shadow test. The choice for an integer-programming model is motivated by the fact that in principle a shadow test may need

more than one item from the same cell in the CxQ to be optimal at the current ability estimate.

To introduce the model, the following notation has to be introduced:

n_{cq}: integer variable for cell (c,q) in table CxQ, that is, n_{cq}=0, 1, ...;

θ_{k-1}: estimate of θ after k-1 items have been administered;

S_{k-1}: set of cells with nonzero decision variable after k-1 items have been administered;

$I_i(\theta)$: Fisher information in the response on item i for an examinee with ability θ;

n_c: number of items in category c to be selected in each CAT;

n_q: number of items in interval q to be selected in each CAT;

φ_{cq}: costs of writing an item for cell (c,q).

The model for the assembly of the shadow test for the selection of the kth item in the CAT simulation can be presented as:

$$\text{Minimize} \sum_c \sum_q \varphi_{cq} n_{cq} \qquad \text{(minimizing costs)} \quad \textbf{(6)}$$

subject to

$$\sum_c \sum_q I_q(\theta_k) n_{cq} \geq T(\theta_k), \; k=1,...,K, \qquad \text{(test information)} \quad \textbf{(7)}$$

$$\sum_c \sum_q n_{cq} = k-1, \; (c,q) \in S_{k-1}, \qquad \text{(previous administered items in the test)} \quad \textbf{(8)}$$

$$\sum_q n_{cq} \geq n_c, \; c=1,...,C, \qquad \text{(categorical constraints)} \quad \textbf{(9)}$$

$$\sum_c n_{cq} \geq n_q, \; q=1,...,Q, \qquad \text{(quantitative constraints)} \quad \textbf{(10)}$$

$$n_{cq} = 0,1,..., \; c \in C, \; q \in Q. \qquad \text{(integer decision variables)} \quad \textbf{(11)}$$

The model has an objective function (Equation 6) for the shadow tests that minimizes a cost function for writing the items in the pool. The information on the

ability parameter at the ability estimate in the CAT is bounded from below by a target value, T, in Equation 7. The constraint in Equation 8 requires the k-1 previously administered items to be in the test. Because of this constraint, the attribute values of these items are automatically taken into account when selecting the kth item. Equation 9 and 10 guarantee that the CAT meets all categorical and quantitative constraints.

Calculating the Blueprint for the Item Pool. In the simulation study, it is counted how often an item in cell (c,q) is administered. The blueprint for the item pool is calculated from these counts. The idea is to calculate the blueprint such that all items will tend to have an equal exposure rate if the item pool is realized and used in operational testing. A uniform item exposure rate is generally considered to be ideal because it prevents both item security problems due to overexposure of a small number of items and loss of resources involved in the writing and pretesting of items that are hardly used at all.

Let the counts be equal to N_{cq} and let M be the maximum number of times an item can be exposed. The blueprint has the following number of items in cell (c,q):

$$I_{cq} = \left\lceil \frac{N_{cq}}{M} \right\rceil . \qquad (12)$$

This formula is justified by the following argument: If the ability distribution in the CAT simulations is a reasonable approximation to the true ability distribution in the population, N_{cq} predicts the number of items needed from cell (c,q). However, to meet the required exposure rates, these numbers should be divided by M. If I_{cq} is not an integer it should be rounded upward.

An empirical example of this method for designing item pools for CAT can be found in Veldkamp and van der Linden (2000).

Computerized Adaptive Testing with Rotating Item Pools.

A different way to deal with item security in CAT is to use a system of rotating item pools. The pools are taken from a master pool and assembled to be parallel. They rotate both over time and locations to prevent examinees from predicting what item pool they will get. The idea of rotating item pools was first described in Stocking and Swanson (1998). The main design question in this section approach is how to design a system of parallel rotating item pools.

In the method of Stocking and Swanson (1998), all the items are assigned to item pools using their weighted deviations model (Swanson & Stocking, 1993). The

objective function in this method is to minimize the differences between the individual item pools and the average item pools with respect to (1) the number of items with the required categorical and quantitative attributes and (2) item information at a selected grid of θ values. The method requires the specification of a set of weights to reflect the importance of the differences. In Stocking and Swanson (1998) this method is applied to get both systems of overlapping and nonoverlapping item pools. The possibility of overlap between item pools is used to permit the selection of underused items in more than one pool. If the exposure rates of the items are known, the number of times they are assigned to an item pool can be used to approximate the ideal of uniform item exposure for all items in the master pool for the population of examinees.

Ariel, Veldkamp, and van der Linden (2002) present a different approach to assembling a system of rotating item pools. Their method is a two-step method based on the values of the item attributes instead of the observed exposure rates of the items. In the first step, items with similar attribute values are assigned to interim sets. In the second step, the items in the interim sets are assigned to item pools. The method was motivated by Gullikşen's (1950) matched random subsets method which was developed to split a test into two parallel halves to estimate a largest lower bound to the test reliability. A formalization of Gulliksen's method based on mathematical programming is given in van der Linden and Boekkooi-Timminga (1988).

We will discuss the method only for the case of the item difficulty and discrimination parameters and a system of two rotating item pools. The generalization to more attributes and larger numbers of item pools follows immediately. Let i and j be any two items in the master pool (i, j=1,..., I). The similarity between i and j is measured by a distance function

$$\delta_{ij} = |a_i - a_j| + w|b_i - b_j| \qquad \text{(distance function)} \quad \textbf{(13)}$$

where a denotes the item discrimination parameters, b the difficulty parameter, and w is a weight factor that can be used to correct for differences in scale between a and b.

4.3.1 Linear Programming Models

To find interim pairs of similar items the following linear programming problem has to be solved:

$$\text{Minimize} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \delta_{ij} * x_{ij} \qquad \text{(objective function)} \quad \textbf{(14)}$$

subject to

$$\sum_{i<j} x_{ij} + \sum_{i>j} x_{ji} = 1, \quad j = 1,...,I \qquad \text{(item pairs)} \quad \textbf{(15)}$$

$x_{ij}=0,1;\ i, j=1,\ldots,I.$ (binary decision variables) **(16)**

The objective function in Equation 14 is minimized to get interim pairs that are as similar as possible. Equation 15 guarantees that every item belongs to only one interim pair.

In the second step of the method, items in the interim pairs, which we denote as Q_r, $r=1,\ldots,R$, are assigned to the two item pools $p=1,2$. To make the pools parallel we want to minimize the difference in information in the pools at a selected grid of ability values, θ_k, $k=1,\ldots,K$.

The decision variables in the model are variables y_{ip} which take the value 1 if item i is assigned to pool p and 0 otherwise. The following model for the assignment of the items is proposed:

minimize z (objective function) **(17)**

subject to

$$\left| \sum_{i \in j} I_i(\theta_k) y_{i1} - \sum_{j \in l} I_j(\theta_k) y_{j2} \right| \le z, \quad k=1,...,K, \qquad \text{(pool information)} \quad \textbf{(18)}$$

$$\sum_{i \in Q_r} y_{ip} = 1, \quad p = 1,2, \quad r = 1,...,R, \qquad \text{(each pool one item from each pair)} \quad \textbf{(19)}$$

$$\sum_{p} y_{ip} = 1, \quad i = 1,...,I. \qquad \text{(each item assigned to one pool)} \quad \textbf{(20)}$$

y_{ip}=0,1. (binary decision variables) **(21)**

The objective function in Equation 18 minimizes the difference in information between the two pools. To guarantee that each pool receives an item from each pair Equation 19 has been included in the model. Equation 20 requires that each item be assigned only once to an item pool.

If overlapping item pools are required, Equation 20 should be replaced by

$$\sum_p y_{ip} = n_i, \quad i = 1,\ldots,I \qquad \text{(item } i \text{ assigned to } n_i \text{ pools)} \quad \textbf{(22)}$$

where $n_i \leq P$ denotes the number of pools item i is assigned to and P is the total number of pools.

Like the previous design models in this paper, the model in Equation 17-21 can be extended with constraints to deal with all possible kinds of categorical and/or quantitative constraints the items pool have to meet.

Empirical Example. An empirical example of the design problem of a rotating item pool system is given. (For the full example, see Ariel, Veldkamp and van der Linden, 2002). The master pool was a previous pool of 2,131 items from the Law School Admission Test (LSAT) all calibrated using the 3-parameter logistic model (Hambleton, Swaminathan & Rogers, 1991). The total number of categorical item attributes led to a content table with nine cells, which we label as C_1, ..., C_9. The number of items per cell ranged from 97 to 371. In addition to the item parameters in the 3-parameter response model, we had word counts per item as a quantitative attribute. Stocking's (1994) recommendation that an item pool have a size approximately 12 times the length of the test was followed in that we assembled a system of four nonoverlapping item pools.

In the first stage, the model in Equation 14-16 was solved using the heuristic approach of simulated annealing. A heuristic was used because the model was too large to find an optimal solution. For a description and application of the technique of simulated annealing, see Veldkamp (1999); for a theoretical introduction, see van Laarhoven and Aarts (1987).

In the second stage, items from the interim sets were assigned to the four nonoverlapping item pools. The software package AIMMS 3.2 (Paragon BV, 2000) was used to solve the model in Equation 17-21. The information in the pools in Equation 19 was constrained at θ_k = -1, 0, 1. Also, the total word count for each item pool was set constraint to a reasonable value close to 20 percent of the total number of words in the master pool.

In order to evaluate the parallelness of the four item pools, CAT simulations from these pools were conducted. In addition, CAT simulations were conducted directly from the master pool. A shadow test approach was used to impose constraints on the CAT with respect to the content attributes, word counts, and length of the CAT. Both the shadow tests and the items picked for administration were selected to maximize the information at the current ability estimate. For each pool, 1,000 examinees were randomly sampled from a standard normal distribution for θ.

The average variances of the estimated ability parameters and the total number of items used in the CAT were used to compare the results (for these criteria, see Hambleton, Swaminathan & Rogers, 1991, p.94). The results are given in Table 2. They show the CAT simulations from the four pools yielded results that were closely enough to consider the pools to be parallel. In addition, the average variance of the ability estimates for each of the four pools was slightly higher than for the master pool, but that the total number of items used from the four pools (612) was dramatically larger than the number used from the master pool. These results are as expected: conducting a CAT directly from the master pool gives us more space to adapt the item selection to the examinees' abilities, but if we move to a system of four pools, the CAT is forced to spread its usage of the pools in much better way.

[Insert Table 2 about here]

Discussion

Typically, item pools are no static entities. In most testing programs, tests are assembled from the pool and new items are pretested and added to the pool on a continuous basis. Hence, two important tasks of item pool management are: (1) monitoring the developments in the item pool; and (2) instructing item writers to write new items to complete the pool. The models in this paper can easily be adapted for use in item pool management. The only thing needed is to correct the decision

variables in the models for the numbers of items and stimuli currently available in the pool.

The principle is illustrated for the model in Equations 1-6. Let ν_{cq} be a constant representing the current number of items in cell (c,q) in the pool and η_{cq} a new decision variable denoting the number of items to be written for this cell. The only adaptation necessary is *substituting* $\nu_{cq}+\eta_{cq}$ for the old decision variables in the model.

If the current items in the pool reveal new patterns of correlation between categorical and quantitative attributes, the cost functions φ_{cq} can be updated by defining them on ν_{cq} rather than the frequencies x_{cq} for the previous item pool, or perhaps on a weighted combination of both. This practice is recommended, for example, if the item writers form a categorical attribute in the definition of table CxQ and new item writers have been hired.

References

Ariel, A., Veldkamp, B.P., van der Linden, W.J. (2002a). *Methods for constructing subpools.* Paper presented at the annual meeting of the Psychometric Society, Chapel Hill, NC.

Ariel, A., Veldkamp, B.P., van der Linden, W.J. (2002b). *Constructing rotating item pools for constrained adaptive testing.* Manuscript submitted for publication.

Boekkooi-Timminga, E. (1991). *A method for designing Rasch model-based item banks.* Paper presented at the annual meeting of the Psychometric Society, Princeton, NJ.

Flaugher, R. (1990). Item Pools. In H. Wainer, *Computerized Adaptive Testing: A primer* (pp. 41-64). Hillsdale, NJ: Lawrence Erlbaum Associates

Gulliksen, H. (1950). *Theory of mental tests.* New York: Wiley.

Hambleton, R.K., Swaminathan H., and Rogers, H.J. (1991). *Fundamentals of item response theory.* California, USA: Sage Publications.

Paragon BV (2000). *AIMMS (Version 3.2)* [Computer software and manual]. Haarlem, The Netherlands: Paragon BV.

Segall, D.O., Moreno, K.E., and Hetter, R.D. (1997). Item pool development and evaluation. In W.A. Sands, B.K. Waters, and J.R. McBride (Eds.). *Computerized adaptive testing: From inquiry to operation.* (pp. 117-130). Washington: Applied Psychological Association.

Stocking, M.L. (1994) *Three practical issues for modern adaptive testing item pools* (ETS Research Report No. 94-5) Princeton, NJ: Educational Testing Service.

Stocking, M.L., & Swanson, L. (1998). Optimal design of item pools for computerized adaptive tests. *Applied Psychological Measurement, 22,* 271-279.

Swanson, L., & Stocking, M.L. (1993). A model and heuristic for solving very large item selection problems. *Applied Psychological Measurement, 17,* 151-166.

van der Linden, W.J. (1998). Optimal assembly of psychological and educational tests. *Applied Psychological Measurement, 22,* 195-211.

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* (pp. 27-52). Boston: Kluwer.

van der Linden, W.J., & Boekkooi-Timminga, E. (1988). A zero-one programming approach to Gulliksen's matched random subtests method. *Applied Psychological Measurement, 12*, 201-209.

van der Linden , W.J., & Reese, L.M. (1998). A model for optimal constrained adaptive testing. *Applied Psychological Measurement, 22,* 259-270.

van der Linden, W.J., & Veldkamp, B.P. (2002). *Constraining item exposure in computerized adaptive testing with shadow tests.* Manuscript submitted for publication.

van der Linden, W.J., Veldkamp, B.P., & Reese, L.M. (2000). An integer programming approach to item bank design. *Applied Psychological Measurement, 24,* 139-150.

van Laarhoven, P.J.M., & Aarts, E.H.L. (1987). *Simulated annealing: theory and applications.* Dordrecht: Reidel.

Veldkamp, B.P. (1999). Multiple objective test assembly problems. *Journal of Educational Measurement, 36,* 253-266.

Veldkamp, B.P., & van der Linden, W.J., (2000). Designing item pools for computerized adaptive testing. In W.J. van der Linden & C.A.W. Glas (Eds.). *Computerized adaptive testing: Theory and practice* (pp. 149-162). Boston: Kluwer.

Way, W.D. (1998). Protecting the integrity of computerized testing item pools. *Educational Measurement: Issues and Practice, 17,* 17-26.

Way, W.D., Steffen, M., & Anderson, G.S. (1998, September). *Developing, maintaining, and renewing the item inventory to support computer-based testing.* Paper presented at the ETS Computer-Based Testing Colloquium, Philadelphia, PA.

Table 2
Results form CAT Simulations

	Average Variance	Number of Items Used
Master Pool	.156	189
Four Item Pools		
Pool 1	.175	149
Pool 2	.181	157
Pool 3	.181	155
Pool 4	.182	151

Titles of Recent Research Reports from the Department of Educational Measurement and Data Analysis. University of Twente, Enschede, The Netherlands.

RR-02-09	B.P. Veldkamp, W.J. van der Linden & A. Ariel, *Mathematical-Programming Approaches to Test Item Pool Design*
RR-02-08	B.P. Veldkamp, *Optimal Test Construction*
RR-02-07	B.P. Veldkamp & A. Ariel, *Extended Shadow Test Approach for Constrained Adaptive Testing*
RR-02-06	W.J. van der Linden & B.P. Veldkamp, *Constraining Item Exposure in Computerized Adaptive Testing with Shadow Tests*
RR-02-05	A. Ariel, B.P. Veldkamp & W.J. van der Linden, *Constructing Rotating Item Pools for Constrained Adaptive Testing*
RR-02-04	W.J. van der Linden & L.S. Sotaridona, *A Statistical Test for Detecting Answer Copying on Multiple-Choice Tests*
RR-02-03	W.J. van der Linden, *Estimating Equating Error in Observed-Score Equating*
RR-02-02	W.J. van der Linden, *Some Alternatives to Sympson-Hetter Item-Exposure Control in Computerized Adaptive Testing*
RR-02-01	W.J. van der Linden, H.J. Vos, & L. Chang, *Detecting Intrajudge Inconsistency in Standard Setting using Test Items with a Selected-Response Format*
RR-01-11	C.A.W. Glas & W.J. van der Linden, *Modeling Variability in Item Parameters in Item Response Models*
RR-01-10	C.A.W. Glas & W.J. van der Linden, *Computerized Adaptive Testing with Item Clones*
RR-01-09	C.A.W. Glas & R.R. Meijer, *A Bayesian Approach to Person Fit Analysis in Item Response Theory Models*
RR-01-08	W.J. van der Linden, *Computerized Test Construction*
RR-01-07	R.R. Meijer & L.S. Sotaridona, *Two New Statistics to Detect Answer Copying*
RR-01-06	R.R. Meijer & L.S. Sotaridona, *Statistical Properties of the K-index for Detecting Answer Copying*
RR-01-05	C.A.W. Glas, I. Hendrawan & R.R. Meijer, *The Effect of Person Misfit on Classification Decisions*
RR-01-04	R. Ben-Yashar, S. Nitzan & H.J. Vos, *Optimal Cutoff Points in Single and Multiple Tests for Psychological and Educational Decision Making*

RR-01-03 R.R. Meijer, *Outlier Detection in High-Stakes Certification Testing*

RR-01-02 R.R. Meijer, *Diagnosing Item Score Patterns using IRT Based Person-Fit Statistics*

RR-01-01 H. Chang & W.J. van der Linden, *Implementing Content Constraints in Alpha-Stratified Adaptive Testing Using a Shadow Test Approach*

RR-00-11 B.P. Veldkamp & W.J. van der Linden, *Multidimensional Adaptive Testing with Constraints on Test Content*

RR-00-10 W.J. van der Linden, *A Test-Theoretic Approach to Observed-Score Equating*

RR-00-09 W.J. van der Linden & E.M.L.A. van Krimpen-Stoop, *Using Response Times to Detect Aberrant Responses in Computerized Adaptive Testing*

RR-00-08 L. Chang & W.J. van der Linden & H.J. Vos, *A New Test-Centered Standard-Setting Method Based on Interdependent Evaluation of Item Alternatives*

RR-00-07 W.J. van der linden, *Optimal Stratification of Item Pools in a-Stratified Computerized Adaptive Testing*

RR-00-06 C.A.W. Glas & H.J. Vos, *Adaptive Mastery Testing Using a Multidimensional IRT Model and Bayesian Sequential Decision Theory*

RR-00-05 B.P. Veldkamp, *Modifications of the Branch-and-Bound Algorithm for Application in Constrained Adaptive Testing*

RR-00-04 B.P. Veldkamp, *Constrained Multidimensional Test Assembly*

RR-00-03 J.P. Fox & C.A.W. Glas, *Bayesian Modeling of Measurement Error in Predictor Variables using Item Response Theory*

RR-00-02 J.P. Fox, *Stochastic EM for Estimating the Parameters of a Multilevel IRT Model*

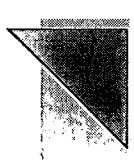
RR-00-01 E.M.L.A. van Krimpen-Stoop & R.R. Meijer, *Detection of Person Misfit in Computerized Adaptive Tests with Polytomous Items*

RR-99-08 W.J. van der Linden & J.E. Carlson, *Calculating Balanced Incomplete Block Designs for Educational Assessments*

RR-99-07 N.D. Verhelst & F. Kaftandjieva, *A Rational Method to Determine Cutoff Scores*

RR-99-06 G. van Engelenburg, *Statistical Analysis for the Solomon Four-Group Design*

Research Reports can be obtained at costs, Faculty of Educational Science and Technology, University of Twente, TO/OMD, P.O. Box 217, 7500 AE Enschede, The Netherlands.



faculty of

EDUCATIONAL SCIENCE AND TECHNOLOGY

A publication by
The Faculty of Educational Science and Technology of the University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

U.S. Department of Education
Office of Educational Research and Improvement (OERI)
National Library of Education (NLE)
Educational Resources Information Center (ERIC)

NOTICE

Reproduction Basis

This document is covered by a signed "Reproduction Release (Blanket)" form (on file within the ERIC system), encompassing all or classes of documents from its source organization and, therefore, does not require a "Specific Document" Release form.

This document is Federally-funded, or carries its own permission to reproduce, or is otherwise in the public domain and, therefore, may be reproduced by ERIC without a signed Reproduction Release form (either "Specific Document" or "Blanket").