ED 308 247 TM 013 667

AUTHOR	Boekkooi-Timminga, Ellen
TITLE	A Cluster-Based Method for Test Construction. Research Report 88-3.
INSTITUTION	Twente Univ., Enschede (Netherlands). Dept. of Education.
PUB DATE	Apr 88
NOTE	40p.; Project funded by the Dutch Organization for the Advancement of Pure Research.
AVAILABLE FROM	Bibliotheek, Department of Education, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands.
PUB TYPE	Reports - Evaluative/Feasibility (142)
EDRS PRICE	MF01/PC02 Plus Postage.
DESCRIPTORS	*Computer Assisted Testing; *Item Banks; Latent Trait Theory; Mathematical Models; *Test Construction; Test Items; Test Selection
IDENTIFIERS	*Cluster Based Test Construction; Information Function (Tests); Integer Programming; Linear Models; Parallel Test Forms

ABSTRACT

A new test construction method based on integer linear programming is described. This method selects optimal tests in small amounts of computer time. The new method, called the Cluster-Based Method, assumes that the items in the bank have been grouped according to their item information curves so that items within a group, or cluster, are interchangeable. Introducing this assumption may reduce the number of decision variables in the model. Because of the assumption of interchangeability, the accuracy of the method will also be reduced, but this reduction is small. The process of test construction is independent of the particular item response model used. The basic model for cluster-based test construction using information functions is described. How the model can be generalized to allow for subject matter constraints is shown. Methods of dealing with additional test specifications and selecting individual test items are described. The algorithm for solving the integer linear programming models is presented. Six test construction problems were then analyzed, and the results of these analyses were presented in three tables. In each case an item bank of 1,000 items was used. These examples revealed that the basic model for cluster-based test construction works well in terms of central processing unit time and accuracy. Additional examples of the test construction method describe the construction process for a selection and a diagnostics test and the process of constructing four parallel tests. Two additional tables give the characteristics of these tests. (SLD)

A Cluster-Based Method for Test Construction

Research Report 88-3

Ellen Boekkooi-Timminga

department of EDUCATION

University of Twente

Division of Educational Measurement and Data Analysis

2

Colofon:

Typing:	Mevr. L.A.M. Padberg
Cover design:	Audiovisuele Sectie TOLAB Toegepaste Onderwijskunde
Printed by:	Centrale Reproductie-afdeling

A Cluster-Based Method for Test Construction

Ellen Boekkooi-Timminga

A cluster-based method for test construction / Ellen Boekkooi-Timminga - Enschede : University of Twente. Department of Education, April, 1988. - 33 pages

Abstract

Recently, some methods for optimal test construction from item banks have been proposed using information functions from IRT. The main problem of these methods is the large amount of time required to identify the optimal test. In this paper, a new approximation method is presented that considers groups of interchangeable items instead of individual items. The method produces accurate results in a small amount of time.

Keywords: Item Response Theory, Test Construction, Linear Programming

A Cluster-Based Method for Test Construction

In 1968 Birnbaum suggested an IRT-based procedure for test construction using information functions. This procedure, which assumes the availability of a calibrated, one-dimensional pool of items, was also used by Lord (1977, 1980). The basic idea is to select those items in the test of which the item information curves fill the area under a target for the test information function. However, neither Birnbaum nor Lord did give a computational procedure for selecting these items.

Recently, some automated methods for item selection based on a target information function approach have been proposed (Boekkooi-Timminga, 1986, 1987, Theunissen, 1985, 1986; van der Linden 1987; van der Linden & Boekkooi-Timminga, 1988). These methods approach test construction from a mathematical programming (in particular, a zero-one linear programming) perspective. The main problem with these methods is the large amount of computer time needed to select the best test items, which is a problem inherent in zero-one programming problems Because of these problems, research on approximations has been conducted. Some approximations have been developed by Boomsma (1986) and Theunissen and Verstralen (1986); however, they are limited to applications of the model proposed by Theunissen (1985). Another approximation method was developed by Adema (1988). This method is applicable to many zero-one programming problems; part of it is used in this paper.

A new test construction method based on integer linear programming is described in this paper that selects optimal tests in small amounts of computer time. This new method, which will be called the cluster-based method, assumes that the items in the bank have been grouped according to their item information curves such that items within a group (cluster) are interchangeable. Introducing this assumption may reduce the number of decision variables in the model drastically. However, because of the interchangeability assumption the accuracy of this new method will also be reduced. In the remainder it will be shown that this reduction is small.

Because of the simplification of the test construction procedure, some of the usual constraints on item selection cannot be met. For instance, inter-item dependencies (Theunissen, 1986) are difficult to handle when the items involved do not belong to the same cluster

This paper first describes the process of item clustering Then, the cluster-based test construction model, the computational procedure, and a few examples are given A discussion concludes the paper.

Item Clustering

The approach to item clustering directly depends on the item response model used. In this paper, it is assumed that the

Rasch–model holds. However, in the discussion of the paper it is indicated how to handle when the two– or three–parameter model has to be assumed.

In the case of the Rasch model, clustering the items is a very simple process. First, the ability scale is partitioned into C equal intervals. Then, all items with difficulty (b) in the same interval are considered as belonging to the same cluster. It is assumed that the ability scale is partitioned within a certain range (e g. –3 to 3); items not falling within this range are included in the outmost clusters. For this procedure, it is very easy to add new items or remove old items from clusters

The mean item information function of a cluster, which will be called the cluster information function, is computed to be used during the item selection process. A simulation study showed that the information function associated with the mean item difficulty b_c of cluster c diffeied very little from the cluster information function For cluster widths of 0 4 logits or less, this deviation from the cluster information value was always less than 1%. The advantage of using the mean item difficulty b_c is that less computational effort is required.

Width of Intervals

An important problem is to determine the appropriate width of the intervals on the ability scale. In order to profit most from the new test construction method, the item bank should consist of as few clusters as possible containing as many items as possible which all can be considered as interchangeable.

To determine the appropriate widths of the intervals a small simulation study was conducted. The percentages of clusters containing more than 10 and 20 items in an item bank of 1000 items were computed. Three item banks with difficulty parameter distributions b~N(0,1), b~N(0,2), and b~U(-3,3)) were simulated with interval widths between 0.5 and 0.05 logits. In view of the requirement that the clusters should consist of as many items as possible, interval widths smaller than 0.2 were dissuaded, especially for item banks with little variance in item difficulty. For instance, in our study a width of 0 1 yielded percentages of clusters with more than 20 items equal to 44.3%, 27.8%, and 18% for the three item banks, respectively. Whereas, the percentages of intervals with more than 10 items were. 54 1%, 67.2%, and 96.7%.

Furthermore, the maximum differences between item information values of items located within the same cluster were computed For interval widths of 0.5, 0.4, 0.3, 0.25, 0.2, 0.1, and 0.05 logits, the maximum differences found were: 6.00%, 3.88%, 2.20%, 1.56%, 1.00%, 0.24%, and 0.08% From these results, it was concluded that interval widths should not exceed 0.4 logits.

A Cluster-Based Test Construction Method

Before the actual test construction process starts a reduction phase is carried out. The purpose of this reduction phase is to exclude items from being selected on basis of characteristics specified by the test constructor Some of these characteristics are: Subject matter, item format, item administration time, item difficulty level, and the number of times items have already been used in other tests (e.g., van der Linden & Boekkooi-Timminga, 1988). It is important to exclude such items because of a reduction of CPU-time and data-storage requirements.

The process of test construction described in this section, is completely independent of the particular item response model used. First, the basic model for cluster-based test construction using information functions is described. Then, it is shown how this model can be generalized in order to allow for subject matter constraints. After the description of the first stage of the test construction procedure, it is described how additional test specifications can be treated in the second stage. Finally, the selection of the individual test items is outlined.

The Basic Model

The model makes use of an objective function described in van der Linden (1987) and van der Linden and Boekkooi-Timminga (1988). The objective function has the advantage of an easy way of deriving the target test information function from a test constructor, because only relative heights of the target function at some freely chosen ability levels need to be specified. Test information is maximized under the condition that this relative distribution is fulfilled (see expressions (1) and (2)). Formally, the target is characterized by a series of lower bounds ($r_1z, \ldots, r_Kz$), in which z is a dummy variable to be maximized, and r_k is the relative information value desired at ability level k.

(1) maximize z

subject to

(2) $$\sum_{c=1}^{C} x_c I_c(\theta_k) - r_k z \geq 0 \qquad k = 1, \ldots, K$$

(3) $$\sum_{c=1}^{C} x_c = N$$

(4) $$x_c \overset{\leq}{\underset{\geq}{=}} n_c \qquad c = 1, \ldots, C$$

(5) $$x_c > 0 \text{ , and integer} \qquad c = 1, \ldots, C$$

(6) $$z \geq 0,$$

The decision variable x_c gives the number of items to be included in the test from cluster c. $I_c(\theta_k)$ is the cluster information value of cluster c at ability level k. Expression (3) represents a constraint fixing the number of items to be selected at N. The maximum, minimum, or exact number of items n_c to be selected from a cluster is specified in (4).

Whereas, x_c is not allowed to take values greater than the number of items included in cluster c (max_c), it is always required that the exact or maximum number n_c is indicated. The constraints in (5) and (6) define the lower bounds and the types of decision variables. In addition to (1) to (6), if the Rasch model applies, constraints on test difficulty can be included in the model. For example, in expression (7) upper and lower bounds B_1 and B_2 are set to the mean item difficulty of the test.

$$(7) \qquad \sum_{c=1}^{C} 1/N\, b_c x_c \begin{matrix} \geq B_1 \\ \leq B_2 \end{matrix}$$

Subject Matter Areas

In most practical situations the subject matter areas covered by the test are of great importantance to the test constructor. Only some small adaptations of model (1) to (7) are needed to deal with constraints on the test contents.

Assume that non-overlapping subject matter areas $j = 1,\ldots,J$ are of interest to the test constructor. Then, the variables x_c and n_c in (1) to (7) can be transformed into x_{cj} and n_{cj}, where x_{cj} defines the number of items on subject matter area j to be selected from cluster c. The maximum, minimum, or exact number of items is given by n_{cj}, where n_{cj} is not allowed to be greater than the total number of items on subject matter area j in cluster c. Furthermore, a summation sign over j has to be added to expressions (2), (3) and (7). Finally, the right-hand sides of expressions (4) and (5) now have to be dependent on j as well.

If it is necessary to control test content with respect to subject matter, it is recommended to constrain the number of items to be selected from each subject matter area. The exact, minimum or maximum numbers to be taken from each area can be defined. Doing so, constraint (3) may become redundant and then may be left out. Expression (8) defines the exact number of items n_j to be selected from subject matter area j, where n_j is not allowed to be greater than the total number of items covering subject matter area j.

$$(8) \qquad \sum_{c=1}^{C} x_{cj} = n_j \qquad\qquad j = 1,\ldots,J$$

Additional Test Specifications

Besides the test specifications dealt with above, possible other specifications can be taken into account in a second stage of the procedure. Examples of such specifications are: Test administration time, item format, and frequency of previous usage of items. From the first test construction stage, it is known how many items have to be included in the test from each cluster. Those clusters from which items need to be selected are further partitioned on basis of the item characteristics to be considered in this second stage. For instance, within these clusters, items may be further partitioned according to their format. If more than one item characteristic needs to be dealt with, the newly formed partitions are partitioned again. For instance, the items in the clusters on item formats may be further partitioned on administration time. Thus, the more item characteristics to be considered, the smaller the groups of items with the same profile of characteristics. Now in this stage of the test construction process again groups of interchangeable items, which will be called item characteristics groups, are considered. Using integer programming it is decided how many items from each of these groups should be included in the test, such that the additional test specifications are fulfilled.

Assume that the basic model in (1) to (6) is considered and that only one additional item characteristic has to be dealt with in this second stage. On the basis of the item characteristic, i = 1, ...,I groups have to be taken into account. Then, decision variable y_{ci} gives the number of items to be selected from item characteristic group i in cluster c. Three groups of constraints to be used here are described below

One group of constraints is always required. It guarantees that the number of items to be taken from each cluster (x_c), as determined in the first stage, is actually included in the test. This is formulated as

$$(9) \qquad \sum_{i=1}^{I} y_{ci} = x_c \qquad\qquad c \in V_c$$

V_c defines the clusters c to be dealt with during the second stage.

Another group of constraints explicitly deals with the newly introduced item characteristics, for instance, constraints setting upper and/or lower bounds to the number of items to be included in the test from item characteristic group i. A maximum m_1, a minimum m_3, and an exact number m_2 of items can be specified by

$$(10) \qquad \sum_{c \in V_c} y_{ci} \begin{matrix} \leq m_1 \\ = m_2 \\ \geq m_3 \end{matrix} \qquad i = 1,\ldots,I$$

Also, for instance, constraints forcing the test administration time to be at least m_4 can be implemented by

$$(11) \qquad \sum_{c \in V_c} \sum_{i=1}^{I} t_{ci} y_{ci} \geq m_4,$$

where t_{ci} gives the mean item administration time for all items within item characteristic group i.

Finally, constraints giving upper and lower bounds to the decision variables y_{ci} are required to guarantee that the numbers of items to be selected from each item characteristic group do not exceed certain values. These constraints can be formulated as follows

$$(12) \qquad \begin{aligned} & y_{ci} \leq n_{ci} && c \in V_c \\ & y_{ci} \geq 0 \text{ and integer valued} && i = 1,\ldots,I \end{aligned}$$

By including dummy variables (d_l) in some of the constraints, the possibility to search for a test with minimal deviations from the desired test properties is provided. For instance, test administration time as close to m_5 as possible can be obtained with

$$(13) \quad \sum_{c \in V_c} \sum_{i=1}^{I} t_{ci} y_{ci} = m_5 + d_1 - d_2.$$

As objective function, a (weighted) sum of the introduced dummy variables is used. A general expression for such an objective function is

$$(14) \quad \text{minimize} \quad \sum_{l=1}^{L} w_l d_l,$$

where L is the total number of dummy variables included in the model. The weights w_i can be used when deviations from certain desired test properties are viewed to be more serious than from others. The only restriction on the d_l's is that they have to take values greater than zero There is a need for this type of objective function, because one is restricted to select items from clusters identified in the previous test construction stage and a solution fitting all constraints not necessarily exists. Using this type of objective function only in incidental cases no solution to the problem is found.

<u>Remark.</u> Two problems occur when the additional test specifications are introduced. First, adding a lot of

different item characteristics implies that the number of decision variables increases rapidly. If this is the case the cluster-based method will be less quick. The other problem is the possibility of not finding a solution. An interactive version of the procedure in which the test constructor can adapt his/her desires with the help of information given by the system is recommended for such cases. The information the test constructor should get from the program includes. (1) An overview of numbers of items in all clusters, each subject matter area, etc, (2) an indication which constraint caused the problem of finding no solution, and (3) a proposal for adapting one or more constraints.

Individual Item Selection

After the numbers of items to be selected from the clusters have been determined the individual items need to be chosen. This can be done by: (1) random item selection, or (2) optimal item selection Random item selection is preferred because of CPU-time advantages However, it is possible to use zero-one programming for optimal selection of the individual items, for instance, minimizing the deviation of the actual test information values from the target information function values.

In the previous stages of the test construction process it was not possible to take into account constraints involving individual items like, for instance: If item i is selected for the test item j and k also have to be selected. When optimal item selection using zero-one programming is applied it is easy to deal with those constraints, using decision variables x_i taking the values 0 if item i is not, and 1 if item i is selected for the test (e.g. van der Linden & Boekkooi-Timminga, 1988; Theunissen, 1986). However, much CPU-time will be needed when using this option, especially if many items have to be considered. In such cases it is recommended not to use this option. If random item selection is used, it is only possible to check whether the test selected fits all constraints after the selection process has been conducted. If the test fits, then it is accepted; if not, another selection has to be made.

Computational Procedure

In this section first the algorithm for solving the integer linear programming models is described. Then, some experiments with this algorithm for six examples of test construction problems are discussed. The experiments include a comparison of objective function values determined by zero-one programming and the cluster-based method. Next, the six test construction problems are solved for three different item banks partitioned in four different ways. Also, the effect of the upper bound n_c (expression (4)) on CPU-time is looked at. Finally, a comparison is made between the actual objective function values computed after the individual items had been selected and those obtained from the cluster-based method considering the cluster information functions.

The Algorithm

The algorithm was based on the branch-and-bound method proposed by Land and Doig (1960). It was adapted because of the amount of computer time inherent in solving integer programming problems (e.g. Lenstra & Rinnooy Kan, 1979). The procedure first computes the solution to the relaxed problem obtained by leaving the integer requirements in expression (5) out of consideration, resulting in an upper bound value z for the integer problem. After the relaxed solution was obtained the fractional values were rounded to the nearest integer value. The rounded solution was accepted if the resulting objective function value did not differ more than 1% from the optimal value of z, and if all constraints were met. Depending on the wishes of the test constructor the 1% norm could be adapted; however, one should take care of the fact that a solution should remain possible.

If the rounded solution could not be accepted, the integer solution was determined following a slightly adapted version of a procedure proposed by Adema (1988). In the procedure decision variables were fixed to zero if the reduced costs were greater than z - .999z. The value of .999 is optional and can be changed according to the wishes of the test designer. After this, the branch-and-bound procedure started. The procedure was finished when an integer solution for which the objective function did not differ more than 1% from the optimal value of z was found.

Experiments

In Tables 1, 2, and 3 the results of some experiments with the above procedure using the basic model in (1) to (6) are summarized. Six test construction problems were analyzed. The test specifications for each problem can be found in Table 1. An item bank consisting of 1000 items with $b \sim N(0,2)$ was used. The interval widths were: 0.4, 0.3, 0.25, and 0.2. The given amount of CPU-time is the time needed only for actual optimization and writing the output file. All experiments were performed on a MS-DOS XT-computer with a clock frequency of 8 MHz.

It is well known for maximization problems that the value of the objective function for the solution to a relaxed zero-one programming problem ($x_i \in [0,1]$) is an upper bound to the value for the zero-one programming problem ($x_i \in \{0,1\}$). A comparison of these upper bounds for the zero-one programming problem and for the cluster-based method indicates the accuracy of the latter.

Insert Table 1 about here

Table 1 summarizes the objective function values obtained and CPU-times needed for the six test construction problems. Only results for interval widths of 0.4 and 0.2 were included in the Table. Furthermore, the differences between the upper bound values for the zero-one programming problems and the objective function values are given in percentages of the upper bound values. It is seen that these differences are small for each problem; only for Problem 1 and 3 differences slightly larger than 1% were found.

The relaxed zero-one problems were solved on a mainframe DEC-2060 computer: the CPU-times also include the time needed to read the input-file. If the approximation method for solving zero-one programming problems by Adema (1988) were used, the CPU-times would have been higher, because this method first computes the relaxed solution. For the cluster-based method the CPU-times were very low. The greatest amount of time was needed for Problem 1 with interval width 0.2. However, in this case the rounded solution was accepted. (Acceptance meant that all constraints were met, and the objective function value did not differ more than 1% from its upper bound z: S(1).) To show the effect of a change of computer, the problem was also solved on an MS-DOS AT-computer with a clock frequency of 15 MHz. The CPU-times for this problem were. 2.04 sec (relaxed). 43 50 (integer), and, 2.09 sec (rounded).

Insert Table 2 about here

In Table 2, the results for each of the six problems with $x_c \leq 10$ are summarized. Also, CPU-times and objective function values for the relaxed integer, and rounded solution are given. It can be seen that only for some try-outs of problem 1 and 3, the rounded solutions were not acceptable. For Problem 1, in some cases the rounded solution did not fit all constraints. All test construction problems were solved in an acceptable amount of CPU-time.

Next, some experiments were carried out to examine the effect of a change in the upper limit n_c ($x_c \leq n_c$) on the CPU-time. An interval width of 0.3 was chosen. Five cases were looked at:

1. $n_c = max_c$,
2. $n_c = 10$,
3. $n_c = max_c/2$ if $max_c \leq 20$, and $n_c = 10$ otherwise,
4. $n_c = max_c/5$ if $max_c \leq 50$, and $n_c = 10$ otherwise. and
5. $n_c = 5$.

It was found that a decrease of n_c generally resulted in an increase of CPU-time For the six test construction problems the minimum and maximum CPU-times (in seconds) for finding the accepted solution were. Problem 1 (22 70-70.40), Problem 2 (5.60-8.90), Problem 3 (3.40-30.60), Problem 4 (3.60-5.70), Problem 5 (1.80-3.90), and Problem 6 (1.30-3.90). Only for Problems 1 and 3 (Case 1), the rounded solutions were not accepted; thus, the corresponding CPU-times were higher.

Having the numbers of items to be selected from each of the clusters, the individual items were selected. For Problems 1 to 6 comparisons were made between the actual objective function values after the individual items had been selected, and the objective function values z obtained from the cluster-based method. The objective function value was chosen to be the larger value of the rounded and first-integer solutions. The widths of the cluster intervals were 0.4, 0.3, 0.25, 0.2. The actual tests were selected in two ways: (1) at random, and (2) such that it worstly reflected the cluster information function. The "worst" test was determined as follows: Two tests were selected to include only items located at the upper or lower ends of the cluster intervals, respectively. Next, only the worst one in terms of z-value was looked at. It was found that random selection almost always resulted in accurate solutions: Except for two cases, the deviations from z were always smaller than 1%. For the worst tests, the deviations were much larger. For Problems 1 and 3 they varied between 2% and 5%; and for Problems 5 and 6 between 0.5% and 3%. Only for Problems 2 and 4 most of these deviations were larger than 5%; the largest deviation was 7 8%. As could be expected. the best results were obtained for the smallest interval widths

Conclusion

From the experiments, it is concluded that the basic model for cluster-based test construction method works well, in terms of CPU-time as well as accuracy. Furthermore, decreasing the width of the cluster intervals causes an increase in the amount of CPU-time needed, because of the larger number of decision variables.

A remarkable observation was the fact that for almost all problems a rounded solution was found that fitted the constraints. which mostly did not deviate more than 1% from the relaxed solution. For Problem 6 it could be observed that the relaxed solution always gave an integer result.

The experiments were also carried out for item banks with difficulty distributions $b \sim N(0,1)$ and $b \sim U(-3,3)$. However, no remarkable differences could be noted, neither in CPU-time needed nor in accuracy.

Examples

In this section two examples of the cluster-based test construction method are given. First, a complete test construction process is described for both a selection and a diagnostics test. Second, the problem of constructing four parallel tests is looked at.

In both examples the same item bank was considered A Rasch item bank of 1000 items with a difficulty distribution of $b \sim N(0,2)$ and a cluster width of 0.25. Furthermore, it was assumed that 25 subject matter areas were covered by the item bank, which were directly related to item difficulty. In Table 3 the distribution of items over subject matter areas and clusters c is given

Insert Table 3 about here

Construction of a Selection and a Diagnostic Test

Two tests were constructed: One for selection (Test A) and the other for diagnostic (Test B) purposes. In case of Test A, one ability point ($\theta = 1$) in which maximal information was needed was of interest; thus, the value of r_k was arbitrarily set at 1. Furthermore, each of the subject matter areas 15 to 20 had to be represented by 5 items in the test.

For Test B the relative information values should be the same at all ability levels $\theta = -1,0,1$. Hence, $r_1 = r_2 = r_3 = 1$. Three items from each of the ten subject matter areas 8 to 10, 12 to 15 and 17 to 19 had to be included in the test. Finally, it was desired that the total test administration time for Test A and B be as close as possible to 150 minutes. The item administration times (in minutes) were obtained through simulation, assuming that they were uniformly distributed over the item bank with $t \sim U(2,12)$.

In Step 1 of the test construction process, the basic model (1) to (6) extended with constraints for the subject matter aspects was used In Step 2 the total test administration time T was taken into account: All clusters where items had to be selected from were partitioned according to their item administration times. Within each cluster, five partitions having mean item administration times of 3, 5, 7, 9, 11 were determined. As objective function $z = h_1 + h_2$ was taken, whereas the constraint for the test administration time was

$$\sum_{c}\sum_{j}\sum_{i} t_i y_{cji} = T + h_1 - h_2,$$

where h_1 and h_2 were dummy variables used in minimizing the deviation from the desired test administration time

Insert Table 4 about here

Table 4 summarizes the characteristics of both tests constructed.

The Construction of Parallel Tests

Tests are considered to be parallel when their information functions are identical (Samejima, 1977). Four tests parallel to the diagnostic test selected in the previous section were determined, excluding the requirement concerning the total testing time. The tests were constructed in two ways: (1) simultaneously, and (2) sequentially.

For simultaneous test construction, the models were adapted slightly: n_c (in this case: n_{cj}) in equation (4) was divided by the number of tests to be constructed After determining the number of items from each cluster, the tests were randomly selected. When the tests were constructed sequentially, the same test construction models were used for each test, adapting n_{cj} after each run. In Table 5 the characteristics of the selected tests are summarized.

Insert Table 5 about here

From Table 5 it can be seen that the tests were more parallel in terms of their information functions when they were constructed simultaneously. Furthermore, less computer time was needed to select them. Another advantage of simultaneous test construction was that the tests were more parallel in terms of subject matter, because for each test the same numbers of items were taken from the same clusters. However, there is also a larger chance of not finding a solution, because the problem is more constrained.

Discussion

In this paper a new procedure for test construction was described. With this method tests fitting the requirements can quickly be selected from large item banks using a micro-computer. The main advantages of the method are the little amounts of CPU-time and data-storage needed.

A critical remark has to be made. When only the first stage of the test construction process is used, the CPU-times will be very low. As long as only a few item characteristics need to be considered in additional test specifications these times will remain low. However, introducing a lot of new item characteristics will make the problem hard, because of the rapid increase of the number of decision variables. When this is the case the advantages of the cluster-based method will be lower.

A final remark regards the use of the two- and three-parameter logistic models instead of the Rasch model. The only difference lies in the process of item clustering. First, a distance measure is needed to reflect the difference between the item information functions of two items. A possible distance measure is the non-overlapping area between the information curves of the pair of items, which can be computed easily by adding successive rectangles of small width between two points. Then, standard procedures can be applied to determine clusters, for instance, using the criterion of a minimal within cluster variance. However, clustering the items in this way will take more time.

Author's Note

The Dutch Organization for the Advancement of Pure Research (Z.W.O.) is gratefully acknowledged for funding this project. This research was conducted while Ellen Boekkooi-Timminga was supported by a PSYCHON-grant of this organization (560-267-001), awarded to Dr. W.J. van der Linden.

I would like to thank Wim J. van der Linden and Jos J. Adema, Department of Education, University of Twente for their helpful comments during the preparation of this manuscript.

References

Adema, J.J. (1988). A note on solving large-scale zero-one programming problems. Enschede, The Netherlands: University of Twente.

Birnbaum A. (1968). Some latent trait models. In: F.M. Lord & M.R Novick, Statistical theories of mental test scores. Reading, Mass.:Addison-Wesley.

Boekkooi-Timminga, E (1986). Algorithms for the construction of parallel tests by zero-one programming. (Research Report 86-3) Enschede, The Netherlands: University of Twente.

Boekkooi-Timminga, E (1987). Simultaneous test construction by zero-one programming. Methodika, 1, in press.

Boomsma, Y. (1986). Item selection by mathematical programming. Enschede, The Netherlands: University of Twente.

Land, A.H. & Doig, A. (1960). An automated method for solving discrete programming problems Econometrica, 28, 497-520.

Lenstra, J.K., & Rinnooy Kan, A.H.G. (1979). Computational complexity of discrete optimization problems. In P.L. Hammer, E.L. Johnson, & B.H. Korte (Eds.), Discrete optimization I. New York: North-Holland Publishing Company.

Lord, F.M. (1977). Practical applications of item characteristic curve theory. Journal of Educational Measurement, 14. 117-138.

Lord, F.M. (1980). Applications of item response theory to practical testing problems. Hillsdale, New Yersey: Lawrence Erlbaum Associates,Inc.

Samejima, F. (1977). A use of the information function in tailored testing. Applied psychological measurement, 1, 233-247.

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

Theunissen, T.J.J.M (1986). Some applications of optimization algorithms in test design and adaptive testing. Applied psychological measurement, 10, 381-389.

Theunissen, T.J.J.M. & Verstralen, H.H.F.M. (1986). Algoritmen voor het samenstellen van toetsen. [Algorithms for test construction], In W.J. van der Linden (Ed.), Moderne methoden voor toetsgebruik en -constructie. Lisse: Swets & Zeitlinger.

van der Linden, W.J. (1987). Automated test construction using minimax programming. In: W.J. van der Linden, IRT-based test construction (Research Report 87-2) Enschede, The Netherlands: University of Twente.

van der Linden, W.J. & Boekkooi-Timminga, E. (1988). A maximin model for test design with practical constraints. Psychometrika, in press.

Table 1
A Comparison of Objective Function Values and CPU-times for the Solutions of the Cluster-Based Approach and the Solution to the Relaxed Zero-One Programming Problem ($b \sim N(0,2)$; $x_c \le max_c$)

Width	z						CPU-time (in sec)		
	S(1)	d(1)	S(2)	d(2)	S(3)	d(3)	S(1)	S(2)	S(3)
Problem 1 $r_k=1$ for $\theta_k=-3,-2,-1,0,1,2,3$; N=40									
0.4	4.1901	(0.25%)	4.1599	(0.97%)	4.1681*	(0.78%)	4.28	23.40	4.28*
0.2	4.1985	(0.06%)	4.1566	(1.05%)	4.1580*	(1.02%)	6.10	142.70	6.15*
——	4.2008						103.87		
Problem 2 $r_k=1$ for $\theta_k=-3,-1,1,3$; N=40									
0.4	4.3481	(0.21%)	4.3298	(0.63%)	4.3298*	(0.64%)	2.36	6.76	2.42*
0.2	4.3563	(0.03%)	4.3161	(0.95%)	4.3442*	(0.30%)	6.98	36.30	7.03*
——	4.3574						114.36		
Problem 3 $r_k=1$ for $\theta_k=-2,0,2$; N=40									
0.4	5.3316	(0.31%)	5.2824*	(1.23%)	5.2554	(1.74%)	1.76	4.73*	1.76
0.2	5.3428	(0.11%)	5.2946*	(1.01%)	5.2136	(2.52%)	3.46	29.93*	3.51
——	5.3484						102.53		
Problem 4 $r_k=1$ for $\theta_k=-1,0,1$; N=40									
0.4	7.8596	(0.03%)	7.8367	(0.32%)	7.8453*	(0.21%)	1.71	3.30	1.71*
0.2	7.8640	(0.03%)	7.8315	(0.39%)	7.8590*	(0.04%)	3.19	6.43	3.24*
——	7.8620						113.25		
Problem 5 $r_k=10$ for $\theta_k=-2,2$ $r_k=1$ for $\theta_k=0$; N=40									
0.4	0.5364	(0.26%)	0.5358	(0.38%)	0.5360*	(0.34%)	0.93	2.64	0.99*
0.2	0.5378	(0%)	0.5374	(0.08%)	0.5370*	(0.16%)	2.42	7.74	2.47*
——	0.5378						69.28		
Problem 6 $r_k=1$ for $\theta_k=0$; N=40									
0.4	9.9993*	(0.08%)	9.9993	(0.08%)	9.9993	(0.08%)	0.66*	0.71	0.71
0.2	10.0000*	(0.09%)	10.0000	(0.09%)	10.0000	(0.09%)	1.48*	1.48	1.48
——	9.9913						42.45		

S(1): Relaxed solution
S(2): Integer solution
S(3): Rounded soltution
d(i): Differences between z for solution S(i) and z for the 0-1 problem in percentages.(i=1,2,3).
—— : Corresponding 0-1 programming problem computed on a Mainframe (DEC-2060) Computer
* : Accepted solution
max_c: Number of items included in cluster c

Table 2
CPU–times and Objective Function Values for the Relaxed, Rounded and Integer Solution of Six Test Construction Problems $b \sim N(0,2)$; $x_c \leq 10$ for all c

Width	CPU–time (in sec)			z			n_f
	S(1)	S(2)	S(3)	S(1)	S(2)	S(3)	
Problem 1							
0.4	7.10	31.90*	7.30	4.1518	4.1143*	4.0737	0
0.3	5.50	22.70*	5.60	4.1955	4.1768*		6
0.25	6.70	39.40*	6.90	4.1984	4.1765*		6
0.2	7.90	167.70	8.40*	4.1985	4.1571*		8
Problem 2							
0.4	2.70	7.20	2.80*	4.3390	4.3191	4.3180*	9
0.3	5.60	33.00	5.70*	4.3500	4.3159	4.3266*	7
0.25	8.30	30.80	8.50*	4.3502	4.3229	4.3357*	11
0.2	10.00	56.40	10.10*	4.3531	4.3132	4.3383*	13
Problem 3							
0.4	2 60	8.40*	2.60	5.3019	5 2673*	5.2089	7
0.3	3.30	8.00	3.40*	5.3229	5.3059	5.2894*	13
0.25	3.80	13.70*	3.90	5.3276	5.2939*	5.2592	15
0.2	5.50	17.90	5.60*	5.3386	5.3180	5.3128*	19
Problem 4							
0.4	4.00	5.70	4.10*	7.6880	7.6752	7.6752*	10
0.3	4.50	6.30	4.60*	7.7681	7.7615	7.7469*	16
0.25	5.40	9.20	5.50*	7.7978	7.7905	7.7577*	18
0.2	6.10	10.20	6.20*	7.8222	7.8112	7.8141*	24
Problem 5							
0.4	2.10	4.40	2.20*	0.5343	0.5340	0.5340*	9
0.3	2 70	4.80	2.70*	0.5344	0.5343	0.5343*	14
0.25	3.10	8.90	3.20*	0.5365	0 5351	0.5359*	16
0.2	3.70	9.30	3.80*	0.5368	0.5366	0.5363*	21
Problem 6							
0.4	1.40*	1.60	1.50	9.4574*	9.4574	9.4574	10
0.3	1.60*	2.10	1.70	9.6832*	9.6832	9.6832	16
0.25	1.90*	2.30	1.90	9.7770*	9 7770	9.7770	20
0.2	2.30*	3.00	2 40	9 8554*	9 8554	9 8554	26

S(1): Relaxed solution
S(2). Integer solution
S(3): Rounded soltution
*: Accepted solution
n_f: Number of decision variables fixed using their reduced costs.

Table 3
Distribution of Subject Matter Areas in an Item Bank consisting of 1000 Items with Difficulty Distribution b~N(0,2) and Width 0.25

Cluster Lower Bound	Subject Matter Areas																									max_c
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	
-3.125	10	6	3																							19
-2.375	2	6	3	1																						12
-2.625		2	7	3	1																					13
-2.375		2	6	8	5	3																				24
-2.125			1	5	10	8	3																			27
-1.875				6	10	12	10	3																		41
-1.625					3	10	12	10	3																	38
-1.375						4	15	16	15	5																55
-1.125							3	15	20	15	3															56
-0.875								5	10	15	10	5														45
-0.625									5	15	19	15	5													59
-0.375										10	15	29	15	10												79
-0.125											6	10	15	15	10	6										62
0.125												10	15	19	15	8	2									69
0.375													10	15	26	15	10									76
0.625														5	10	22	10	5								52
0.875															5	15	23	15	5							63
1.125							(max_{cj})									5	10	18	10	5						48
1.375																	6	6	15	8	5					40
1.625																		3	6	18	8	3				38
1.875																			2	5	12	5	3			27
2.125																					4	10	5			19
2.375																						2	8	3		13
2.625																							1	4	1	6
2.875																							4	5	10	19
max_j	12	16	20	23	29	37	43	49	53	60	53	69	60	64	66	71	61	47	38	36	29	20	21	12	11	

max_c: Maximum number of items to be selected from cluster c
max_j: Maximum number of items to be selected from subject matter area j
max_{cj}: Maximum number of items to be selected from subject matter area j within cluster c

Table 4
Characteristics of Two Tests with Selection and Diagnostic Purposes

	CPU-time (in sec)		z^*	z'	T
	Step 1	Step 2			
Selection	4.70@	7.30@	7.481	7.447	174
Diagnosis	17.70@	34.80@	5.747	5.585	150

z^* : z for the accepted solution
z' : z for a randomly selected test
T : Total test administration time in minutes
@ : Rounded solution fitted all requirements and was accepted

Table 5
Four Parallel Tests Constructed Simultaneously and Sequentially

	CPU-time (in sec)	Test	z^*	z'
Simultaneously	26.80	1	5.655@	5.562
		2	5.655@	5.560
		3	5.655@	5.564
		4	5.655@	5.561
Sequentially	17.70	1	5.747@	5.690
	19.20	2	5.709@	5.615
	17.70	3	5.638@	5.546
	18.90	4	5.624@	5.522

z^*: z for the accepted solution
z': z computed for a randomly selected test
@ : Rounded solution fitted all requirements and was accepted

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

RR-87-1	R. Engelen, Semiparametric estimation in the Rasch model
RR-87-2	W.J. van der Linden (Ed.), IRT-based test construction
RR-87-3	R. Engelen, P. Thommassen, & W. Vervaat, Ignatov's theorem: A new and short proof
RR-87-4	E. van der Burg, & J. de Leeuw, Use of the multinomial jackknife and bootstrap in generalized nonlinear canonical correlation analysis
RR-87-5	H. Kelderman, Estimating a quasi-loglinear models for the Rasch table if the number of items is large
RR-87-6	R. Engelen, A review of different estimation procedures in the Rasch model
RR-87-7	D.L. Knol & J.M.F. ten Berge, Least-squares approximation of an improper by a proper correlation matrix using a semi-infinite convex program
RR-87-8	E. van der Burg & J de Leeuw, Nonlinear canonical correlation analysis with k sets of variables
RR-87-9	W.J. van der Linden, Applications of decision theory to test-based decision making
RR-87-10	W.J. van der Linden & E. Boekkooi-Timminga, A maximin model for test design with practical constraints

RR-88-1	E. van der Burg & J. de Leeuw, Nonlinear redundancy analysis
RR-88-2	W.J. van der Linden & J.J. Adema, Algorithmic test design using classical item parameters
RR-88-3	E. Boekkooi-Timminga, A cluster-based method for test construction

Research Reports can be obtained at costs from Bibliotheek, Department of Education, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands.

Department of
EDUCATION

A publication by
the Department of Education
of the University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands