Methods are proposed for the construction of weakly parallel tests, that is, tests with the same test information function. A mathematical programming model for constructing tests with a prespecified test information function and a heuristic for assigning items to tests such that their information functions are equal play an important role in the methods. The MI and MIDI methods are proposed for constructing tests with a prespecified test information function applying the Minimax model. Similar methods, MANI and MADI, are provided for construction of a weakly parallel test approximately equal with respect to the Maximin criterion. The four methods were applied on a real item bank of 600 items from college placement mathematics tests (520 items were from 13 previously administered American College Testing Assessment Program tests, and 80 were from the College Mathematics Placement Program). The numerical examples indicated that the tests were constructed quickly and that the heuristic gave good results. However, the heuristic was not applicable for every set of practical constraints (i.e., constraints with respect to test administration time, test composition, or dependencies between items). Four tables and four graphs present information about the constructed tests.

(Author/SLD)
Methods and Models for the Construction of Weakly Parallel Tests

Jos J. Adema

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

This document has been reproduced as received from the person or organization originating it.
Minor changes have been made to improve reproduction quality.

Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
J. NELISSEN

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

University of Twente

Division of Educational Measurement and Data Analysis
Methods and Models for the Construction of Weakly Parallel Tests

Jos J. Adema
Methods and models for the construction of weakly parallel tests, Jos J. Adema - Enschede: University of Twente, Department of Education, September, 1990. - 36 pages
Computerized test construction by mathematical programming is possible given the availability an item bank calibrated under an IRT model. In this paper methods are proposed for the construction of weakly parallel tests, that is, tests with the same test information function. In the methods a mathematical programming model for constructing tests with a prespecified test information function and a heuristic for assigning items to tests such that their information functions are equal play an important role. Numerical examples show that the tests are constructed very fast and that the heuristic gives good results. However, the heuristic is not applicable for every set of practical constraints, such as, constraints with respect to test administration time, test composition, or dependencies between items.

Keywords: Item Banking, Test Construction, Weakly Parallel Tests, Mathematical Programming, Heuristics.
In this paper an item bank is defined as a large collection of items calibrated under an item response model. Given an item bank, tests can be constructed automatically by the application of mathematical programming models (Adema & van der Linden, 1989; Baker, Cohen & Barmish, 1988; Boekkooi-Timminga, 1989; van der Linden & Boekkooi-Timminga, 1989; Theunissen, 1985). The goal of mathematical programming models is to optimize an objective function under a number of constraints. In mathematical programming models for test construction the goal is often to maximize the quality of the test under the constraint that the composition of the test is as required by the test constructor. A measure for the quality of a test is the test information function, because the information function for a maximum likelihood estimator of ability is the reciprocal of the (asymptotic) sampling variance of the estimator (Lord, 1980). Therefore, the test information function plays an important role in most mathematical programming models for test construction. Under the condition of local independence, the information function of a test can be computed by addition of the item information functions:

\[ I(\theta) = \sum_{i=1}^{N} I_i(\theta), \]
where $N$ is the number of items in the test, $\theta$ the ability parameter, and $I_i(\theta)$ the information function of item $i$.

Boekkooi-Timminga (1987, 1990) has proposed mathematical programming models for the construction of weakly parallel tests, where according to Samejima (1977) two weakly parallel test forms are "a pair of tests which measure the same ability and whose test information functions are identical".

Two main parts of this paper can be distinguished. In Part A it is assumed that a test information function is available and one or more tests should be constructed such that all tests have the same information function. In the following section two methods are described for solving this problem. In Part B of the paper two methods are given for selecting weakly parallel tests that are optimal with respect to the Maximin criterion. In this part first the criterion is formulated. Then, two methods are proposed for solving the problem. Next, some numerical examples are worked out to give an impression of the practicality of the proposed methods with respect to CPU time and accuracy.

**PART A: The Construction of Weakly Parallel Tests with a Prespecified Information Function**

In this part it is shown how one or more tests with a prespecified test information function can be constructed. In the first section the case of one test is considered. In the
last two sections methods are given for the construction of more than one test. These methods make use of the Minimax Model that is presented in the first section.

A Minimax Model for the Construction of Weakly Parallel Tests

In this section a mathematical programming model for the selection of items is given. It produces tests that approximate a prespecified test information function best and in addition guarantees that, for example, the number of items in the test and the composition of the test are as specified by the test constructor. Such a model can be used to select a test that is weakly parallel to a given test.

The test information function is considered at a number of ability levels $\theta_k$, $k = 1, \ldots, K$. The test constructor can choose the number and spacing of these ability levels. Let $I_i(\theta_k)$ be the information value of item $i$ at ability level $\theta_k$ and $T(\theta_k)$ the test information required at ability level $\theta_k$. The decision variables $x_i$ in the mathematical programming model, denoting if an item is or is not selected for the test, are defined as follows:

$$x_i = \begin{cases} 0 & \text{item } i \text{ not in the test} \\ 1 & \text{item } i \text{ in the test}. \end{cases}$$
The decision variables $u_k$ denote the difference between $T(\theta_k)$ and the test information function value at $\theta_k$ for the case that the latter is smaller than $T(\theta_k)$:

$$u_k = \max \{0, T(\theta_k) - \sum_{i=1}^{I} I_i(\theta_k)x_i\}.$$ 

If the test information function value at $\theta_k$ is larger than $T(\theta_k)$, the difference between the two is given by $v_k$:

$$v_k = \max \{0, \sum_{i=1}^{I} I_i(\theta_k)x_i - T(\theta_k)\}.$$ 

The decision variable $w$ is equal to the absolute value of the largest gap in information function value over all ability levels:

$$w = \max \{ u_k + v_k \}$$

$$= \max \{ \max u_k; \max v_k \}.$$ 

Given these definitions of the decision variables, the Minimax Model is formulated as:

(1) minimize $w,$

subject to
The objective of the model is to minimize the largest difference for all \( \theta_k \) between the given test information function and the information function of the test to be constructed. The objective function (1) and the constraints (2) imply that the decision variable \( w \) is equal to the largest difference between \( T(\theta_k) \) and the test information function at ability level \( \theta_k \) for all \( \theta_k \). By constraints (3) \( u_k - v_k \) is equal to \( T(\theta_k) \) minus the test information function value at \( \theta_k \). The constraints presented in (4) are a general notation for practical constraints, that is, constraints with
respect to the composition of the test, the administration time of the test, etc. (see van der Linden & Boekkooi-Timminga, 1993). The number of items in the test is set equal to N by constraint (5).

Method MI
The construction of, say, P weakly parallel tests can be done by applying model (1)-(6) P times. Every time the model has been applied, the selected items are deleted from the item bank so that no item is contained in more than one test. Throughout this paper, this method of constructing weakly parallel tests will be called MI. As can be seen in the numerical examples below sometimes the psychometric quality of the tests will decrease in the order in which they are constructed by method MI. In the next section a method is described that does not have this drawback.

Method MIDI
This section is addressed to the problem of simultaneous construction of P weakly parallel tests with a prespecified test information function. The problem is solved here in two steps that will now be described. In the first step one large test is constructed that is P times the size of each of the tests to be constructed. In the second step the items are assigned to the P tests such that the selected tests have the same composition and test information function. In the second step the heuristic DIFMIN is applied. This heuristic will be
Weakly Parallel Tests

described here for the case of practical constraints that imply a partition of the item bank. Consider for example, a mathematical item bank that can be partitioned into geometry and intermediate algebra items; one may determine that a test should contain 20 geometry and 20 intermediate algebra items. For other types of constraints DIFMIN should be adapted or is not applicable.

DIFMIN is based on known heuristics for solving the "makespan" scheduling problem (e.g., Coffman, Lueker & Rinnooy Kan, 1988). In this problem jobs have to be distributed among two or more parallel machines so as to minimize the time needed to process them. A good heuristic for the "makespan" scheduling problem according to probabilistic analysis (see, e.g., Bruno & Downey, 1986; Frenk & Rinnooy Kan, 1985; Loulou, 1984) is LPT (Largest Processing Time). In LPT the jobs are sorted in order of decreasing processing time and, next, in decreasing order of processing time, each job is assigned to the machine for which the sum of the processing times of the jobs already assigned is smallest. In our problem the tests can be seen as machines and the items as jobs. Each item, however, does not have one characteristic but K, namely the information function values at \( \theta_k \), \( k = 1, \ldots, K \). This along with the practical constraints and the fact that the number of items should be equal for each test make the construction of weakly parallel tests more difficult than the "makespan" scheduling problem. As in LPT the heuristic starts with sorting the
Weakly Parallel Tests

items, that is, for each ability level $\theta_k$ the items are sorted in order of decreasing information function value. Then, the items are assigned to the tests on a one by one basis. In each iteration of DIFMIN to each test an item is assigned under the restriction that these items be from the same subset of the partition of the item bank. The goal for each assignment is to close the largest gap in information function value for all tests and ability levels.

In the description of the heuristic, $Q$ denotes the $Q$-th iteration. After $N$ iterations DIFMIN stops.

**Step 1:** Use Minimax (1) - (8) to construct a test that is $P$ times the size of each of the $P$ tests to be constructed with $P*\text{T}(\theta_k)$ instead of $\text{T}(A,)$ in (3), $P*N$ instead of $N$ in (5) and $P*b_j$ instead of $b_j$ in (4).

**Step 2.** (Heuristic DIFMIN)

**Step 2a:** For each ability level, $\theta_k$, sort the items selected in Step 1 in order of decreasing information value.

**Step 2b:** Initialization: The first item in the list of $\theta_1$ is assigned to the first test ($p=1$). Set $Q$ equal to 1.

**Step 2c:** Successively for tests $p = 2$ to $P$: Choose the item having the most information at $\theta_1$, belonging to the same part of the item bank as the item selected in Step 2b, and not already assigned, and assign it to test $p$.

**Step 2d:** If the number of items in each test is equal to $N$, that is, if $Q = N$, then STOP, else let $Q := Q+1$ and compute.
Weakly Parallel Tests

\[
\max(k) = \max_{p} I_p(\theta_k), \quad k = 1, \ldots, K,
\]

where \( I_p(\theta_k) \) is the sum of the information function values at \( \theta_k \) of the items already selected for test \( p \).

Compute the ability level \( \theta_{\min} \) and test \( p_{\min} \) which has the largest gap in information function value with respect to all tests and ability levels:

\[
\max(\min) - I_{p_{\min}}(\theta_{\min}) = \max_{\min} \max_{p} \{\max(k) - I_p(\theta_k)\}.
\]

The first item in the list of \( \theta_{\min} \) not yet assigned to a test is assigned to \( p_{\min} \).

**Step 2e:** Compute \( \theta_{\min} \) and \( p_{\min} \) such that

\[
\max(\min) - I_{p_{\min}}(\theta_{\min}) = \max_{\min} \max_{p \in \pi_{Q-1}} \{\max(k) - I_p(\theta_k)\}
\]

where \( \pi_{Q-1} \) is the set of tests with \( Q-1 \) items. Select the first item in the list of \( \theta_{\min} \) that has not already been selected for a test and belongs to the same part of the item bank as the last item selected in Step 2d. Repeat this step until all tests contain \( Q \) items.

**Step 2f:** Go to Step 2d.
Weakly Parallel Tests

Example:
Suppose 6 items are selected in Step 1 and that these items should be assigned to two tests such that they have (approximately) the same test information function at ability levels \( \theta_1 \) and \( \theta_2 \). The data are given by:

<table>
<thead>
<tr>
<th>Item</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>( I(\theta_1) )</td>
<td>0.8</td>
<td>1.1</td>
<td>2.0</td>
<td>0.7</td>
<td>0.5</td>
<td>1.4</td>
</tr>
<tr>
<td>( I(\theta_2) )</td>
<td>0.6</td>
<td>0.9</td>
<td>0.4</td>
<td>0.8</td>
<td>_</td>
<td>1.1</td>
</tr>
<tr>
<td>Subset</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>G</td>
<td>IA</td>
<td>IA</td>
</tr>
</tbody>
</table>

where G stands for geometry and IA for intermediate algebra. The following lists are the result of Step 2a:

\( \theta_1 \): 3 6 2 1 4 5
\( \theta_2 \): 5 6 2 4 1 3.

According to Step 2b item 3 is assigned to test 1. Also, item 2 is assigned to test 2 and not item 6 (Step 2c), because items 3 and 6 belong to different subsets. The computations in Step 2d imply that the most informative item at \( \theta_1 \) should be selected for test 2. Item 3 was already selected; thus, item 6 is selected for test 2. Because item 5 is the only item belonging to the same subset as item 6, this item is selected for test 1 (Step 2e). After execution of Step 2e, Step 2d is executed again. The next item to be assigned should give the most information at ability level \( \theta_2 \) and is...
selected for test 1. This implies that item 4 is assigned to test 1. Only item 1 remains and it is assigned to test 2. The results are as follows:

<table>
<thead>
<tr>
<th>Test</th>
<th>Items</th>
<th>$T(\theta_1)$</th>
<th>$T(\theta_2)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3,4,5</td>
<td>3.2</td>
<td>2.4</td>
</tr>
<tr>
<td>2</td>
<td>1,2,6</td>
<td>3.3</td>
<td>2.6</td>
</tr>
</tbody>
</table>

PART B: The Construction of Weakly Parallel Tests under the Maximin Criterion

In this part two methods based on MI and MIDI are described for the construction of weakly parallel tests, where the tests are optimal with respect to the Maximin criterion. This criterion states that a test should give as much information as possible under the restriction that the test information function has the shape required by the test constructor and that the number of items is fixed. This criterion was proposed by van der Linden and Boekkooi-Timminga (1989). As opposed to Part A, the test constructor now specifies a relative instead of an absolute target test information function. A brief introduction of the Maximin Model for the construction of one test will be given below.

**Maximin Model**

The test information function is considered at a number of ability levels $\theta_k$, $k = 1, \ldots, K$. The number and spacing of
these levels should be specified by the test constructor. The relative shape of the target test information function should be specified by the test constructor by choosing \( r_k \), \( k = 1, \ldots, K \). Let \( y \) be a decision variable such that \((r_1y, \ldots, r_Ky)\) is a series of lower bounds to the test information function. If \( N \) is the prescribed number of items in the test, then the Maximin Model is formulated as follows:

\[
\begin{align*}
\text{(9)} & \quad \text{maximize } y, \\
\text{subject to} \\
\text{(10)} & \quad \sum_{i=1}^{I} I_i(\theta_k)x_i - r_ky \geq 0, \quad k = 1, \ldots, K, \\
\text{(11)} & \quad \sum_{i=1}^{I} x_i = N, \\
\text{(12)} & \quad x_i \in \{0, 1\}, \quad i = 1, \ldots, I, \\
\text{(13)} & \quad y \geq 0.
\end{align*}
\]

The lower bounds are forced to be as high as possible by maximizing \( y \) in the objective function (9). The constraints (10) are formulated such that \((r_1y, \ldots, r_Ky)\) is a series of lower bounds to the test information function. The number of items in the test is controlled by constraint (11).
In the next two sections methods are proposed for the construction of weakly parallel tests, which are optimal with respect to the Maximin criterion.

Method MAMI

If the test information function of the weakly parallel tests is known the tests can be selected by applying method MI. Method MAMI is based on the same idea. Two steps can be distinguished in the method. In the first step an estimate of the test information function is computed. In the second step method MI is applied.

Step 1: A good approximation of the test information function is computed by constructing a large test with the Maximin Model that is $P$ times the size of each of the $P$ tests. By dividing the information function values of the large test at the specified ability levels by $P$, a target test information function is found for the Minimax Model.

Step 2: By applying the Minimax Model $P$ times each time deleting items selected at an earlier stage the $P$ weakly parallel tests are constructed (Method MI).

Example:

An item bank consists of 400 items. The first 200 items are of the multiple-choice type and the other 200 items are of the matching type. A test constructor wants to select items
for 3 weakly parallel tests each with 20 multiple-choice and 20 matching items. S(he) needs tests that measure well in the ability range [-2,2] and, therefore, the test information function is specified at the ability levels $\theta = -2$, $\theta_2 = 0$, and $\theta_3 = 2$, and the corresponding values of $r_k$ are $r_1 = r_2 = r_3 = 1$. The Maximin Model is formulated as:

$$\text{maximize } y,$$

subject to

$$\sum_{i=1}^{400} I_i(\theta_k) x_i - y \geq 0, \quad k = 1, 2, 3,$$

$$\sum_{i=1}^{200} x_i = 60,$$

$$\sum_{i=201}^{400} x_i = 60,$$

$$x_i \in \{0,1\}, \quad i = 1, \ldots, 400,$$

$$y \geq 0.$$

Let the information function values of the large test be $L(\theta_k), k = 1, 2, 3$; then the target test information function
is $T(\theta_k) = L(\theta_k)/3$. Thus, the following Minimax Model should be solved three times:

$$\begin{align*}
\text{minimize} & \quad w, \\
\text{subject to} & \quad u_k + v_k \leq w, \\
& \quad \sum_{i \in I_k} I_i(\theta_k)x_i + u_k - v_k = T(\theta_k), \\
& \quad \sum_{i \in I_k} x_i = 20, \\
& \quad \sum_{i \in I_{MC}} x_i = 20, \\
& \quad x_i \in \{0,1\}, \\
& \quad u_k, v_k \geq 0, \\
& \quad w \geq 0.
\end{align*}$$

Every time a test is constructed the items already selected are deleted from the item bank. Subset $I$ is the set of items not deleted from the item bank. Subsets $I_M$ and $I_{MC}$ are sets of items of the matching and multiple-choice type not deleted from the item bank.
Method MADI

The difference between method MIDI and MADI is that in the latter not the Minimax Model but the Maximin Model is used in Step 1.

Step 1: As in method MAMI, method MADI starts with the construction of a test by the Maximin Model that is P times the size of each of the weakly parallel tests.

Step 2: Next, the items selected for the large test are assigned to the P tests by the heuristic DIFMIN such that weakly parallel tests are created.

Example:
In this example the same problem as in the last example is considered. First, a large test is constructed as in method MAMI. Heuristic DIFMIN then assigns the selected items to the tests such that all three tests consist of 20 multiple-choice and 20 matching items and the test information functions are approximately equal.

Numerical Examples

In the examples below mathematical programming models had to be solved several times. The application of exact algorithms is too time consuming, therefore a heuristic has
been used. The applied heuristic solution strategy is in all cases so-called optimal rounding (van der Linden & Boekkooi-Timminga, 1989), because the practical constraints in the examples are such that always a feasible suboptimal solution will be found. Optimal rounding in this sense means, that first the relaxed model \((0 \leq x_i \leq 1\) instead of \(x_i \in (0,1)\)) is solved, then all variables \(x_i\) equal to 0 or 1 are fixed and, finally, the reduced model is solved to optimality by the branch-and-bound method (Land & Doig, 1960). The optimal rounding method was implemented in ECL control programs for the software package MPSX/370 V2. ECL is a computer language based on PL/1. All methods considered in this paper were implemented in control programs such that they could be executed in one run. The execution times of the control programs are displayed in the tables below to show the practicability of the methods. The runs were executed on an IBM9370 computer.

The item bank used in the examples is an existing Academic College Testing (ACT) item bank and was described by Ackerman (1989). The bank consisted of 600 items; 520 items were from 13 previously administered ACT Assessment Program (AAP) tests and 80 were from the Collegiate Mathematics Placement Program (CMPP). The items were calibrated using the 3-parameter model (Birnbaum, 1968). The bank was divided into six content areas:

1. Arithmetic and Algebraic Operations (AAO);
2. Arithmetic and Algebraic Reasoning (AAR);
(3) Geometry (G);
(4) Intermediate Algebra (IA);
(5) Number and Numeration Concepts (NNS);
(6) Advanced Topics.

From the bank items were selected to create weakly parallel tests with 40 items (4 AAO items, 14 AAR items, 8 G items, 8 IA items, 4 NNS items, and 2 AT items).

Methods MI and MIDI

The Minimax Model (1) - (8) can be applied to construct one or more weakly parallel tests with a fixed target test information function. The methods MI and MIDI described above have been developed for this case. In this section the ability levels and the target information function values (θ_k, T(θ_k)) are (-1.6, 2.0), (-0.8, 5.4), (0.1, 12.1), (0.8, 21.3), (1.6, 10.8), and (2.4, 3.1), respectively.

The results for MI are displayed in Table 1. The number of tests P is equal to 6. The test information function values are given for all the tests.

Insert Table 1 here

Figure 1 shows the 6 test information functions.
Weakly Parallel Tests

After the first 5 tests were constructed there was a lack of "good" items. Therefore, the last test information function having the lowest top, is not as close to the target as the information functions of the other tests.

In Table 2 the test information functions of the tests constructed by MIDI are shown for P ranging from 2 to 6.

An illustration of the test information functions for P = 6 is given in Figure 2.

The item bank contains a few items with extremely high difficulty. Therefore, it is possible again that the test information functions increase for ability levels outside the range considered. The tests were constructed simultaneously by MIDI and, therefore, there is no decrement in quality of
the tests in the order of selection. Most of the CPU time was needed for the first step in the method. This step was more time consuming for \( P = 6 \) than for \( P \) ranging from 2 to 5. Thus, the total CPU time was larger for \( P = 6 \).

**Methods MAMI and MADI**

Two methods have been described for the construction of weakly parallel tests which are optimal with respect to the Maximin criterion: MAMI and MADI. In this section the ability levels and the relative information function values \((\theta_k, r_k)\) are \((-1.6, 2.0), (-0.8, 5.4), (0.0, 12.1), (0.8, 21.3), (1.6, 10.8), \) and \((2.4, 3.1)\), respectively.

The test information function values are shown in Table 3 for the method MAMI for values of \( P \), 2 through 6.

The test information functions for \( P = 6 \) are displayed in Figure 3.
Weakly Parallel Tests

For all values of $P$, ranging from 2 to 6 the information function of the last test constructed was considerably worse than the information function of the other tests. The aberrant information function in Figure 3 is the information function of the last test.

Table 4 shows the same results as Table 3, but now for MADI.

Figure 4 displays the information functions for $P = 6$.

Comparing Figures 3 and 4, it can be seen that MADI does not suffer the drawback that later constructed tests tend to be worse than earlier constructed tests. This is because the $P$ tests are constructed simultaneously.
Discussion

In this paper the methods MI and MIDI were proposed for constructing tests with a prespecified test information function. Similar methods namely MAMI and MADI were given for the construction of weakly parallel test approximately equal with respect to the Maximin criterion. The four methods were applied on a real item bank. In the next two sections the practicality of the methods is discussed.

Methods MI and MIDI

The methods MI and MIDI can be used for the construction of weakly parallel tests with a fixed target test information function. Both methods can construct tests in a couple of minutes from a bank of 600 items. Thus, CPU time is no obstacle in using the methods. The method MI has the drawback that tests are selected sequentially, so that tests constructed later tend to be worse than earlier constructed tests. In Table 1 it is seen that this problem occurs for P = 6. The information function of the sixth test is not as good as the one of the other five tests in the closeness of the match of test information function and target information function. For the MIDI method the test information functions are closer to the target test information function than for the MI method, because the tests are constructed simultaneously. MIDI is also faster than MI. In MIDI the heuristic DIFMIN is used and this heuristic is described here.
only for the case of practical constraints that imply a partition of the item bank. The drawback of MIDI is that the method is not applicable for other types of practical constraints.

Methods MAMI and MADI

The methods MAMI and MADI can be used to construct weakly parallel tests with test information functions (approximately) optimal with respect to the Maximin criterion. If these methods are compared with respect to CPU time, it is clear that MADI is faster than MAMI. However, both methods can construct weakly parallel tests in a few minutes. The last test constructed by MAMI has always by far the worst test information function, because the "good" items have already been selected. In this respect MAMI is not as good as MADI. Method MADI, however, suffers the same drawback as MIDI.
References


Acknowledgement

The author wishes to thank Sjoerd Baas and Noud Gademann for their helpful comments, ACT and Terry Ackerman for making the item bank available to me, and Lorette Bosch for her assistance in preparing the figures.
### Table 1

Information Functions for Weakly Parallel Tests Constructed by Method MI

<table>
<thead>
<tr>
<th>p</th>
<th>θ₁</th>
<th>θ₂</th>
<th>θ₃</th>
<th>θ₄</th>
<th>θ₅</th>
<th>θ₆</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.936</td>
<td>5.285</td>
<td>12.073</td>
<td>21.073</td>
<td>10.991</td>
<td>2.971</td>
</tr>
<tr>
<td>2</td>
<td>1.805</td>
<td>5.256</td>
<td>12.218</td>
<td>21.430</td>
<td>10.841</td>
<td>2.902</td>
</tr>
<tr>
<td>3</td>
<td>2.037</td>
<td>5.788</td>
<td>12.456</td>
<td>21.568</td>
<td>10.389</td>
<td>3.072</td>
</tr>
<tr>
<td>4</td>
<td>2.051</td>
<td>5.368</td>
<td>12.061</td>
<td>21.166</td>
<td>10.674</td>
<td>3.033</td>
</tr>
<tr>
<td>5</td>
<td>1.898</td>
<td>5.371</td>
<td>12.278</td>
<td>21.446</td>
<td>10.717</td>
<td>2.853</td>
</tr>
</tbody>
</table>

**Note.** Total CPU-time: 1.97 minutes.
### Table 2

**Information Function for Weakly Parallel Tests Constructed by Method MIDI**

<table>
<thead>
<tr>
<th>P</th>
<th>( \theta_1 )</th>
<th>( \theta_2 )</th>
<th>( \theta_3 )</th>
<th>( \theta_4 )</th>
<th>( \theta_5 )</th>
<th>( \theta_6 )</th>
<th>CPU (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1.976</td>
<td>5.137</td>
<td>11.992</td>
<td>21.530</td>
<td>10.755</td>
<td>3.122</td>
<td>0.55</td>
</tr>
<tr>
<td></td>
<td>1.979</td>
<td>5.397</td>
<td>12.024</td>
<td>21.457</td>
<td>10.796</td>
<td>3.144</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2.039</td>
<td>5.583</td>
<td>12.281</td>
<td>21.518</td>
<td>10.742</td>
<td>3.054</td>
<td>0.63</td>
</tr>
<tr>
<td></td>
<td>1.958</td>
<td>5.588</td>
<td>12.043</td>
<td>21.313</td>
<td>10.767</td>
<td>2.929</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.945</td>
<td>5.438</td>
<td>12.094</td>
<td>21.549</td>
<td>10.859</td>
<td>2.990</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2.030</td>
<td>5.480</td>
<td>11.980</td>
<td>21.314</td>
<td>10.693</td>
<td>2.962</td>
<td>0.51</td>
</tr>
<tr>
<td></td>
<td>2.149</td>
<td>5.328</td>
<td>11.987</td>
<td>21.357</td>
<td>10.959</td>
<td>3.130</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>2.080</td>
<td>5.351</td>
<td>12.169</td>
<td>21.447</td>
<td>10.716</td>
<td>3.106</td>
<td>0.63</td>
</tr>
<tr>
<td></td>
<td>1.938</td>
<td>5.528</td>
<td>12.093</td>
<td>21.248</td>
<td>10.757</td>
<td>2.963</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.976</td>
<td>5.352</td>
<td>12.122</td>
<td>21.266</td>
<td>10.874</td>
<td>3.235</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.986</td>
<td>5.339</td>
<td>12.024</td>
<td>21.345</td>
<td>10.553</td>
<td>3.100</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.968</td>
<td>5.327</td>
<td>11.986</td>
<td>21.371</td>
<td>10.833</td>
<td>3.033</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1.943</td>
<td>5.497</td>
<td>12.107</td>
<td>21.235</td>
<td>10.772</td>
<td>3.102</td>
<td>0.86</td>
</tr>
<tr>
<td></td>
<td>2.146</td>
<td>5.507</td>
<td>12.083</td>
<td>21.156</td>
<td>10.709</td>
<td>3.101</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.074</td>
<td>5.382</td>
<td>12.072</td>
<td>21.200</td>
<td>10.817</td>
<td>3.073</td>
<td></td>
</tr>
</tbody>
</table>
Table 3

Information Functions for Weakly Parallel Tests Constructed by Method MAMI

<table>
<thead>
<tr>
<th>P</th>
<th>θ₁</th>
<th>θ₂</th>
<th>θ₃</th>
<th>θ₄</th>
<th>θ₅</th>
<th>θ₆</th>
<th>CPU (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3.078</td>
<td>8.722</td>
<td>18.369</td>
<td>31.670</td>
<td>16.529</td>
<td>4.689</td>
<td>1.50</td>
</tr>
<tr>
<td>3</td>
<td>2.818</td>
<td>7.790</td>
<td>17.241</td>
<td>30.079</td>
<td>15.734</td>
<td>4.714</td>
<td>1.94</td>
</tr>
<tr>
<td></td>
<td>2.847</td>
<td>8.085</td>
<td>16.976</td>
<td>30.364</td>
<td>15.892</td>
<td>4.345</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.889</td>
<td>6.875</td>
<td>16.469</td>
<td>29.144</td>
<td>15.044</td>
<td>4.458</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2.496</td>
<td>7.293</td>
<td>16.318</td>
<td>28.241</td>
<td>14.976</td>
<td>4.433</td>
<td>2.16</td>
</tr>
<tr>
<td></td>
<td>2.609</td>
<td>7.028</td>
<td>16.135</td>
<td>28.128</td>
<td>15.165</td>
<td>4.508</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.859</td>
<td>5.904</td>
<td>17.034</td>
<td>26.648</td>
<td>16.219</td>
<td>4.044</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.551</td>
<td>6.862</td>
<td>15.732</td>
<td>26.583</td>
<td>14.100</td>
<td>4.244</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.544</td>
<td>7.060</td>
<td>15.521</td>
<td>26.479</td>
<td>13.971</td>
<td>4.477</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.657</td>
<td>5.264</td>
<td>17.482</td>
<td>24.353</td>
<td>15.923</td>
<td>4.032</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>2.163</td>
<td>6.795</td>
<td>15.085</td>
<td>24.751</td>
<td>14.133</td>
<td>3.923</td>
<td>2.73</td>
</tr>
<tr>
<td></td>
<td>2.480</td>
<td>7.133</td>
<td>15.384</td>
<td>25.713</td>
<td>13.753</td>
<td>3.159</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.505</td>
<td>7.586</td>
<td>15.175</td>
<td>25.451</td>
<td>14.365</td>
<td>3.280</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.429</td>
<td>6.969</td>
<td>15.586</td>
<td>25.066</td>
<td>14.080</td>
<td>4.001</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.189</td>
<td>6.840</td>
<td>15.466</td>
<td>25.398</td>
<td>14.283</td>
<td>3.517</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.440</td>
<td>4.268</td>
<td>17.616</td>
<td>22.458</td>
<td>13.198</td>
<td>4.134</td>
<td></td>
</tr>
</tbody>
</table>
Table 4

Information Functions for Weakly Parallel Tests Constructed by Method MAUI

<table>
<thead>
<tr>
<th>P</th>
<th>θ₁</th>
<th>θ₂</th>
<th>θ₃</th>
<th>θ₄</th>
<th>θ₅</th>
<th>θ₆</th>
<th>CPU (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3.107</td>
<td>8.727</td>
<td>18.184</td>
<td>31.873</td>
<td>16.583</td>
<td>4.955</td>
<td>0.63</td>
</tr>
<tr>
<td></td>
<td>2.881</td>
<td>8.838</td>
<td>18.417</td>
<td>32.013</td>
<td>15.586</td>
<td>5.050</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2.792</td>
<td>7.835</td>
<td>17.276</td>
<td>30.388</td>
<td>15.453</td>
<td>4.624</td>
<td>0.61</td>
</tr>
<tr>
<td></td>
<td>2.723</td>
<td>7.881</td>
<td>17.155</td>
<td>29.942</td>
<td>15.834</td>
<td>4.397</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2.909</td>
<td>7.467</td>
<td>16.308</td>
<td>28.193</td>
<td>14.587</td>
<td>4.186</td>
<td>0.65</td>
</tr>
<tr>
<td></td>
<td>2.510</td>
<td>7.245</td>
<td>15.685</td>
<td>28.020</td>
<td>15.620</td>
<td>3.820</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.705</td>
<td>7.111</td>
<td>15.897</td>
<td>27.973</td>
<td>14.575</td>
<td>4.715</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>2.532</td>
<td>6.783</td>
<td>15.920</td>
<td>27.515</td>
<td>14.149</td>
<td>4.657</td>
<td>0.65</td>
</tr>
<tr>
<td></td>
<td>2.280</td>
<td>7.177</td>
<td>15.625</td>
<td>26.482</td>
<td>14.343</td>
<td>3.580</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.504</td>
<td>6.913</td>
<td>15.256</td>
<td>26.474</td>
<td>15.174</td>
<td>3.600</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>2.262</td>
<td>6.863</td>
<td>15.790</td>
<td>25.343</td>
<td>13.869</td>
<td>3.544</td>
<td>0.70</td>
</tr>
<tr>
<td></td>
<td>2.352</td>
<td>7.130</td>
<td>15.288</td>
<td>24.977</td>
<td>14.538</td>
<td>3.544</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.248</td>
<td>7.234</td>
<td>15.998</td>
<td>25.204</td>
<td>14.402</td>
<td>3.545</td>
<td></td>
</tr>
</tbody>
</table>
Figure 1. Information functions of tests constructed by method MI for $P = 6$.

Figure 2. Information functions of tests constructed by method MIDI for $P = 6$.

Figure 3. Information functions of tests constructed by method MAMI for $P = 6$.

Figure 4. Information functions of tests constructed by method MADI for $P = 6$. 
Titles of recent Research Reports from the Division of
Educational Measurement and Data Analysis.
University of Twente, Enschede,
The Netherlands.

RR-90-4  J.J. Adema, Methods and Models for the Construction of Weakly Parallel Tests
RR-90-3  H.J. Vos, Simultaneous Optimization of Classification Decisions Followed by an End-of-Treatment Test
RR-90-2  H. Tobi, Item Response Theory at subject- and group-level
RR-90-1  P. Westers & H. Kelderman, Differential item functioning in multiple choice items
RR-89-6  J.J. Adema, Implementations of the Branch-and-Bound method for test construction problems
RR-89-5  H.J. Vos, A simultaneous approach to optimizing treatment assignments with mastery scores
RR-89-4  M.P.F. Berger, On the efficiency of IRT models when applied to different sampling designs
RR-89-3  D.L. Knol, Stepwise item selection procedures for Rasch scales using quasi-loglinear models
RR-89-2  E. Boekkooi-Tinndnga, The construction of parallel tests from IRT-based item banks
RR-89-1  R.J.H. Engelen & R.J. Jannarone, A connection between item/subtest regression and the Rasch model
RR-88-18  H.J. Vos, Applications of decision theory to computer based adaptive instructional systems
RR-88-17  H. Kelderman, Loglinear multidimensional IRT models for polytomously scored items
RR-88-16  H. Kelderman, An IRT model for item responses that are subject to omission and/or intrusion errors
RR-88-15  H.J. Vos, Simultaneous optimization of decisions using a linear utility function
RR-88-14  J.J. Adema, The construction of two-stage tests
RR-88-13  J. Kogut, Asymptotic distribution of an IRT person fit index
RR-88-12  E. van der Burg & G. Dijksterhuis, Nonlinear canonical correlation analysis of multiway data

42
RR-88-11  D.L. Knol & M.P.F. Berger, Empirical comparison between factor analysis and item response models
RR-88-10  H. Kelderman & G. Macready, Loglinear-latent-class models for detecting item bias
RR-88-9   W.J. van der Linden & L.J.H.M. Eggen, The Rasch model as a model for paired comparisons with an individual item parameter
RR-88-8   R.J.H. Engelen, W.J. van der Linden, & S.J. Oosterloo, Item information in the Rasch model
RR-88-7   J.H.A.N. Rikers, Towards an authoring system for item construction
RR-88-6   H.J. Vos, The use of decision theory in the Minnesota Adaptive Instructional System
RR-88-5   W.J. van der Linden, Optimizing incomplete sample designs for item response model parameters
RR-88-4   J.J. Adema, A note on solving large-scale zero-one programming problems
RR-88-3   E. Boekkooi-Timminga, A cluster-based method for test construction
RR-88-2   W.J. van der Linden & J.J. Adema, Algorithmic test design using classical item parameters
RR-88-1   E. van der Burg & J. de Leeuw, Nonlinear redundancy analysis

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