ED 450 130 TM 032 320

AUTHOR	Veldkamp, Bernard P.
TITLE	Modifications of the Branch-and-Bound Algorithm for Application in Constrained Adaptive Testing. Research Report.
INSTITUTION	Twente Univ., Enschede (Netherlands). Faculty of Educational Science and Technology.
REPORT NO	RR-00-05
PUB DATE	2000-00-00
NOTE	30p.
AVAILABLE FROM	Faculty of Educational Science and Technology, University of Twente, TO/OMD, P.O. Box 7500 AE Enschede, The Netherlands.
PUB TYPE	Reports - Research (143)
EDRS PRICE	MF01/PC02 Plus Postage.
DESCRIPTORS	*Adaptive Testing; Algorithms; *Computer Assisted Testing; *Test Construction; Test Items
IDENTIFIERS	*Branch and Bound Method; Constraints; *Mathematical Programming

ABSTRACT

A mathematical programming approach is presented for computer adaptive testing (CAT) with many constraints on the item and test attributes. Because mathematical programming problems have to be solved while the examinee waits for the next item, a fast implementation of the Branch-and-Bound algorithm is needed for this approach. Eight modifications of the algorithm especially designed for application in CAT are described. In order to investigate the effects of the modifications, two empirical examples using simulation were studied. The modified Branch-and-Bound algorithm selected the items in the adaptive tests in a realistic amount of time, while the resulting tests met the constraints. (Contains 2 tables, 1 figure, and 17 references.) (Author/SLD)

Modifications of the Branch-and-Bound Algorithm for Application in Constrained Adaptive Testing

Research Report
00-05

Bernard P. Veldkamp

faculty of
EDUCATIONAL SCIENCE AND TECHNOLOGY

University of Twente

Department of
Educational Measurement and Data Analysis

Modifications of the Branch-and-Bound Algorithm for Application in Constrained Adaptive Testing

Bernard P. Veldkamp

Abstract

A mathematical programming approach for computer adaptive testing (CAT) with many constraints on the item and test attributes is presented. Because in the approach, mathematical programming problems have to be solved while the examinee waits for the next item, a fast implementation of the Branch-and-Bound algorithm is needed. Eight modifications of the algorithm specifically designed for application in CAT are described. In order to investigate the effects of the modifications, two empirical examples were studied. The modified Branch-and-Bound algorithm selected the items in the adaptive tests in a realistic amount of time, while the resulting tests met the constraints.

Introduction

In Computer Adaptive Testing (CAT), a number of approaches have been proposed to deal with content constraints on adaptive tests. Both Kingsbury and Zara (1991) and Segall, Moreno, and Hetter (1997) suggest to partition the item pool and to use an algorithm to select items from these pools during the CAT or even to use separate adaptive tests and to combine final scores afterwards. In Stocking and Swanson (1993) the items are selected sequentially, minimizing the weighted sum of deviations from the constraints. Recently, a new approach to constrained CAT based on the assembly of shadow tests prior to item selection, was introduced in van der Linden and Reese (1998).

The following pseudo algorithm for this Shadow Test Approach is given in van der Linden (2000):

Step 1: Initialize the ability estimator.

Step 2: Assemble a shadow test that meets the constraints and has maximum information at the current ability estimate.

Step 3: Administer the item in the shadow test with maximum information at the ability estimate.

Step 4: Update the ability estimate.

Step 5: Return all unused items to the pool.

Step 6: Adjust the constraints to allow for the attributes of the item administered.

Step 7: Repeat Steps 2-6 until m items have been administered.

If the item pool is well designed, the shadow test approach guarantees that all constraints are met for every CAT assembled. In order to gain these profits an integer programming formulation of the test assembly problem in Step 2 is necessary.

Several algorithms have been proposed for solving integer programming problems in test assembly, for example, those in Contest (Timminga, van der Linden & Schweizer, 1996), the heuristic for the Weigthed Deviation Model (Swanson & Stocking, 1993) and several implementations of the Network-Flow algorithm by Armstrong, Jones & Wang (1995). These algorithms have proved to be very useful in test assembly, but they may result in occasional constraint violation, or in a sub-optimal solution. So, when they are applied it is not always possible to gain full profits of the shadow test approach. Alternatively, a Branch-and-Bound (B&B) algorithm can be used to find the optimal solution (Adema, 1992, Timminga, van der Linden, and Schweizer, 1996). However, the B&B algorithm is rather time consuming. Especially for large adaptive tests or tests with an item-set structure, this might be a problem since in CAT only a small amount of time is available for selecting the next item.

The goal of the present research is to find modifications of the B&B algorithm such that the shadow test approach can be used for the problem of assembling adaptive tests with large number of constraints from large item pools. In this paper, first a general formulation of a mathematical programming model for selecting the k-th item in an adaptive test as well as the B&B algorithm is given. Then, eight modifications of the algorithm applicable in a CAT-framework are described. To investigate the effect of these modifications, two test assembly problems were studied, and their results are presented. Finally, the modifications are evaluated and some recommendations about their use are given.

Mathematical Programming Model

In order to apply mathematical programming techniques for solving the problem of selecting the next item, the problem has to be written in a specific format. An objective function that will be optimized, has to be defined and the test specifications have to be written as constraints. Let,

$I_i(\widehat{\theta}_{k-1})$ be the information provided by item i at the estimated ability level $\widehat{\theta}_{k-1}$,

N be the number of items in the item pool,

$i = 1, ..., N$ be the index ranging over all items,

S be the number of sets in the pool,

$s = 1, ..., S$ be the index ranging over all item sets,

x_i and y_s denote 0-1 decision variables for selecting item i or item set s,

$n_s^{\min}$ be the minimum number of items in set s to be selected when s is in the test,

$n_s^{\max}$ be the maximum number of items in set s to be selected when s is in the test,

V_g be a subset of indices of sets with attribute g,

V_j be a subset of indices of items with attribute j,

V_s be a subset of indices of items in set s,

k be the index ranging over the items in the adaptive test,

S_{k-1} be the set of indices of $k-1$ item already selected,

a_s, b_i be values of the set attributes and the item attributes,

n_g, n_j be attribute values of the entire test,

T_l be the fixed total test length.

Model

The model for selecting the k-th item in the adaptive test can be formulated as follows:

$$\max \sum_{i=1}^{N} I_i\left(\widehat{\theta}_{k-1}\right) x_i \qquad \text{(maximize information function)} \qquad (1)$$

subject to:

$$\begin{aligned}
\sum_{i=1}^{N} x_i &= T_l && \text{(test length)} && \text{(2a)}\\
\sum_{s \in V_g} a_s y_s &\leq n_g && \text{(constraints at set level)} && \text{(2b)}\\
\sum_{i \in V_j} b_i x_i &\leq n_j && \text{(constraints at item level)} && \text{(2c)}\\
\sum_{i \in S_{k-1}} x_i &= k-1 && \text{(items already presented in the test)} && \text{(2d)}\\
n_s^{\min} y_s &\leq \sum_{i \in V_s} x_i \leq n_s^{\max} y_s && \text{(number of items selected for set s)} && \text{(2e)}\\
x_i, y_s &\in \{0,1\}. && && \text{(2f)}
\end{aligned}$$

B&B Algorithm

It is important to notice that both variables x_i and y_s are 0-1 variables. So, the B&B algorithm can be applied. Intuitively, the algorithm can be described in the following manner. The B&B algorithm starts with the computation of the optimal solution to the relaxation, i.e., the problem where the integer constraints in Equation 2f are left out. The problem is now in the class of linear programming problems, which are much easier to solve. The computation is done by the simplex method, a well known method for solving linear programming problems (Winston, 1994). Since the set of constraints of the relaxation is smaller than and contained within the set of the original problem, the objective value of the relaxation will be higher, and it can be used as an upper bound $\overline{z}$ for the solution of the test assembly problem. When the solution of the relaxation only has integer values, the optimal solution to the test assembly problem is already obtained. If not, the problem is branched in two new problems. In one problem a non-integer variable is fixed at zero. In the other this variable is fixed at one. These new problems are treated the same way, until all branches are searched. Whenever a solution is infeasible, i.e., conflicts with the constraints, or when the value of the relaxation ($\overline{z}$) is smaller than the lower bound ($\underline{z}$), i.e., the best integer solution found so far, this branch is pruned. The reason is that it is no longer possible to meet the constraints or to find a better solution, because some variables x_i have been fixed. The remaining problems are branched by fixing new variables, and bounds for the new problems are calculated. When a better integer solution is found, the value of $\underline{z}$ is updated. The B&B method terminates when all branches have been investigated or pruned.

Example

To illustrate the B&B algorithm, a small test assembly problem is introduced. The objective function in this problem is to optimize Fisher's information function at a certain ability estimate. Three constraints are involved. Item 1 and 2 are in the same content class. From this content class at least one item should be in the test. The word count for the total test should be less or equal to 150. Finally, since Item 2 and Item 3 contain clues to each other, only one of them can be in the test. The 0-1 LP formulation of this problem is given in Equation 3 through Equation 7:

$$\max 0.805x_1 + 1.158x_2 + 0.753x_3 \tag{3}$$

subject to:

$$\begin{aligned} x_1 + x_2 &\geq 1, & (4)\\ 80x_1 + 79x_2 + 68x_3 &\leq 150, & (5)\\ x_2 + x_3 &\leq 1, & (6)\\ x_i &\in \{0,1\}. & (7)\end{aligned}$$

The B&B search is presented in Figure 1. The upper half of a node consist of a problem number and the objective value of the relaxation; the lower half contains the solution of the relaxed problem. The optimal solution is in Node 4, which has a feasible solution with the highest value for the objective function.

==================

Insert Figure 1 at about here

==================

Pseudo Algorithm

The B&B method can be described by the following pseudo algorithm.

Step 1. Select a node and solve the relaxation associated with it.

Step 2. When the value of the relaxation is smaller than the lower bound for the optimal solution, no better solution can be found in this branch. Prune the branch. Go to Step 6.

Step 3. When the value of the relaxation is greater than the lower bound, but the values of x_i are not integer, the branch might contain an improvement. Go to Step 5.

Step 4. When the solution of the relaxation is feasible and is better than the best solution found so far, adjust the lower bound. Delete all nodes with upper bounds lower than the new lower bound. Go to step 6.

Step 5. Branch the node. In one branch the value of x_i is set equal to $x_i = 0$. In the other branch $x_i = 1$.

Step 6. Stop when all branches are investigated or pruned, otherwise go to Step 1.

A formal description of the algorithm is given in Nemhauser & Wolsey (1988, page 355).

In the next section eight modifications of this general algorithm are described.

Modifications of the B&B algorithm

The B&B algorithm is in the class of implicit enumeration methods. Algorithms in this class search the set of possible integer solutions in such a way that not all individual possibilities need be considered. Nevertheless their worst-case performances sometimes are considerably time consuming. This feature of the B&B algorithm is especially unfavorable for CAT, where the time available for selecting the next item is limited. In this context, accuracy and speed are essential to apply an algorithm. Therefore, measures to improve the performance of the B&B algorithm were investigated. From the theory of integer programming several improvements were derived. Eight of such improvements are described below.

1. Starting Solution

The first modification is the use of a starting solution. In the general algorithm no starting solution is used and the lower bound is set equal to $\underline{z} = -\infty$. Whenever an integer solution to the problem is known, it can be used as a starting solution and this solution will provide a better initial lower bound.

This observation can be used for speeding up the B&B algorithm. As already noted in Step 2, a branch is of no interest and can be pruned, when the upper bound of the objective values of solutions in the subset is smaller than a lower bound to the objective value of the optimal solution. Using a good starting solution might provide a strong lower bound and hence a lot of branches might be pruned. So fewer branches have to be investigated and the algorithm finishes earlier (see also Adema, 1992).

Especially for CAT, this feature might be of interest. When the shadow test approach (van der Linden and Reese, 1998) is used and the constraints do not depend on the estimated theta $\widehat{\theta}$, the solution to the problem in Iteration $k-1$ can be used as a starting solution to the problem in Iteration k, because the constraints in both problems are generally identical. After a number of items, the ability estimate of the examinee stabilizes and the starting solution even provides stronger lower bounds. First remind that the problem in Iteration k is based on the ability estimate $\widehat{\theta}_{k-1}$, and the problem in Iteration $k-1$ is based on the ability estimate $\widehat{\theta}_{k-2}$. When the ability estimate stabilizes after a number of items, $\widehat{\theta}_{k-2}$ is not much different from $\widehat{\theta}_{k-1}$. Therefore, the objective function in Iteration $k-1$ is not much different from the objective function in Iteration k, and since the constraints in both iterations are generally equal, the solution to the problem in Step $k-1$ provides a good approximation to the solution of the problem in step k.

When no prior information about the examinee is known, the initial estimate is often set equal to $\widehat{\theta}_0 = 0$ for all examinees. For this initial estimate, the first item in a CAT can be calculated in advance. Once available, it can be used as a starting solution for all examinees. When a variable initial ability estimate is used, a different strategy should be followed. Instead of calculating one starting solution in advance, starting solutions should be calculated for a number of points on the ability scale $\widehat{\theta}_{0j}$, $j = 1..J$, that cover a sufficiently large part of the ability range. The solution for $\widehat{\theta}_{0j}$ closest to the initial estimate of the examinee, θ_0, can be used as a starting solution.

2. Integer slack variables

A second improvement (Williams, 1990, page 204) deals with formulation of the constraints. Consider an arbitrary constraint $\sum a_i x_i \leq b$, where a_is and b are integer constants and $x_i \in \{0,1\}$. An example is the word count constraint given in Equation 5, where a_i represent the number of words per item, and b is the maximum number of words in the test. Other examples are time-limit constraints, content constraints, constraints on gender-orientation of the items, or on item format.

Because the variables x_i are integer, the difference $b - \sum a_i x_i$ will also be integer. So, it is possible to introduce an integer slack variable δ, and reformulate the constraint as

$$\sum a_i x_i + \delta = b. \tag{8}$$

During the B&B search, one could also branch on these integer slack variables. When applied to the problem in Equation 1 through Equation 2f, two versions of this modification are needed:

$$\begin{aligned} \sum_{s \in V_g} a_s y_s + \delta_g &= n_g & (9) \\ \sum_{i \in V_j} b_i x_i + \delta_j &= n_j & (10) \end{aligned}$$

Both at set level and at item level some constraints can be rewritten as equalities. There is advantage in introducing integer slack variables and giving them priority in the branching process (see next modification). This idea is due to Mitra (1973).

3. Priority ordering

A general recommendation concerning B&B methods is that one should branch on variables with the greatest impact first (Williams, 1990, page 205). Applying this recommendation to the general algorithm involves a more detailed specification of Step 5. In Step 5 the set of possible solutions is split in two parts by fixing a variable. The algorithm should be modified such that a node is divided into two branches by fixing a variable with high priority.

The gain involved in this modification can be illustrated by considering an assembly problem where both set and item variables are present. In this case, difference between the impact of their variables exists. Fixing item variables has little effect, but fixing set variables forces the algorithm to select items from the sets that are chosen to be in the test. Also, if set variables are fixed at zero, all item variables that belong to these sets are automatically fixed at zero. A different example deals with the integer slack variables introduced above. Integer slack variables δ often have greater impact on the problem than the other variables in the constraint. In a content constraint, the slack variable determines the number of items or number of sets with a certain content to be in a test.

4. Branching strategy

Now that a priority order can be made, Step 5 of the algorithm could be specified more in detail. Variables with high priority should be chosen first. However, the question remains which variable to choose in the same priority class. In modern IP-solvers some options are provided (ILOG, 1999). The solution of the LP-relaxation x_{RP}, calculated in Step 1, is often used as an indicator. In this relaxation, the variables do not have to take integer values, so the variables x^i_{RP} can be written as the sum of an integer and a fractional part, $x^i_{RP} = i^i_{RP} + f^i_{RP}$, where $0 \leq f^i_{RP} \leq 1$.

The Maximum Infeasibility Rule. This rule branches on the variable whose fractional value f^i_{RP} deviates most from zero and one. In this way, all variables get integer values slowly but surely.

The Minimum Infeasibility Rule. This rule branches on the variable whose fractional value f^i_{RP} deviates less from zero or one. The idea is to find a good integer solution to the problem in a small amount of time. This solution provides a strong lower bound and branches are pruned earlier.

More Intelligent Branching Rules. Rules, like branching on pseudo costs or on pseudo reduced costs (Adema, 1992, Forrest, Hirst and Tomlin, 1974, Mitra, 1973), can also be applied. These rules generally need more time to find a sharp lower bound, but often perform better over all.

Automatic Selection of Branching Variable. This rule is a combination of the above described rules. An algorithm determines which rule seems most promising and decides which rule is applied.

In CAT, a choice should be made between the intelligent branching rules and the minimum infeasibility rule. The maximum infeasibility rule will probably take to much time. If time is very limited the minimum infeasibility rule will probably perform better, because its first shot often provides a good solution. If a little more time is available the intelligent branching rule is more likely to produce the best result. Its first shot may be worse than for the minimum infeasibility rule but it will find improvements faster and eventually result in a better solution.

5. Node Selection

In Step 1 of the general algorithm, a node is selected to be investigated. Instead of randomly choosing a node, more advanced strategies can be used. The depth-first strategy, the best-bound strategy and the best-estimate strategy (Nemhauser and Wolsey, 1988, pages 358-359, e.g.) are reasonable alternatives.

The Depth-First Strategy. This strategy selects the most recent node. Whenever this strategy is used, a different variable is fixed in every iteration. In this way, an integer solution is found in a small amount of time. Unfortunately, in combination with the use of a priority ordering, this strategy has a serious drawback. Due to the priority ordering, variables with high priority are fixed first and variables with lower priorities last. After finding the first integer solution, the most recent problems result from fixing a low-priority variable. In the subsequent iterations, only the values of the variables with low priority are changed. So both modifications counteract.

The Best-Bound Strategy. This strategy selects the node with best optimal value for the relaxed problem. Whenever the relaxed problem provides a good upper bound, the difference between this upper bound and the lower bound is a good indication of the improvement that can be reached by selecting this node.

The (Alternative) Best-Estimate Strategy. This strategy selects a node by estimating its best integer solution. This estimate is made by correcting for the fractional values of the variables x^i_{RP} (e.g. see Mitra, 1973, Nemhauser & Wolsey, 1988).

In a CAT program with item sets, some variables have higher priorities than others. Therefore, the depth-first strategy is unsuitable for obtaining results in a small amount of time. Both remaining strategies have their pro's and con's. The best-bound strategy has a time advantage, because the objective values of the relaxations are known. On the other hand, the relaxed problem does not provide a very good bound. The best-estimate strategy performs better in selecting promising problems, but more time is needed for calculating the estimates. So, no theoretical arguments can be offered to prefer one strategy over the other.

6. Special Ordered Sets (SOS)

The next modification deals with a special kind of constraints. A special ordered set of Type 1 (Beale and Tomlin, 1969) is a set of variables from which at most one can be equal to one:

$$\sum_{j=1}^{k} x_j \leq 1. \tag{11}$$

When these kind of constraints are in a problem, the fifth step of the algorithm can be made more efficiently (e.g., Forrest, Hirst and Tomlin, 1974). In Step 5 of the algorithm a node is branched by fixing a variable at $x_i = 0$ or at $x_i = 1$. However, instead of branching a node by fixing a single variable, it is possible to fix half the number of items in the special ordered set. First split the variables $x_j, j = 1, ..., k$ into two subsets x_j^1, where $j = 1, ..., f$ and x_j^2,where $j = f + 1, ..., k$. Since only one variable in the entire set can be equal to $x_i = 1$, this variable is either in the first half or in the second half of this set. So, either

$$\begin{aligned} x_j &= 0 \qquad j = 1, ..., f, \\ \sum_{j=f+1}^{k} x_j &\leq 1. \end{aligned}$$

or

$$\begin{aligned} \sum_{j=1}^{f} x_j &\leq 1, \\ x_j &= 0 \qquad j = f + 1, ..., k. \end{aligned}$$

In this way the node can be branched, and half the number of variables, rather than one, is fixed in the same iteration.

This modification of the fifth step can be applied in several ways. In a test assembly context these kind of constraints appear, for example, when a single item has to be chosen from a set.

7. Fixing Variables with Near-Integer Values

Some ideas to speed up the B&B algorithm were proposed by Crowder, Johnson and Padberg(1983). One of them deals with fixing variables. For a pure zero-one linear programming problem, it can be shown that the following implications hold: After Step 1, the objective value of the relaxation, z^i_{RP}, the continuous optimal reduced costs d_j for all items $j = 1, ..., N$, and a true lower bound $\underline{z}$ are known. Reduced costs are the amounts by which the coefficient of an unselected item in the objective function must be improved before this item is selected in the test in an optimal solution of the LP. In Crowder Johnson and Padberg (1983) the following theorem was proved.

The values of z^i_{RP}, d_j, and $\underline{z}$ can be used to fix variables of unselected items at the value 0 or 1:

(1) fix x_j to 0 if in the solution of the relaxation $x_j = 0$ and $z^i_{RP} - \underline{z} < d_j$;

(2) fix x_j to 1 if in the solution of the relaxation $x_j = 1$ and $z^i_{RP} - \underline{z} < -d_j$.

In Adema, Boekkooi-Timminga, and van der Linden (1991) this idea is applied, but instead of $\underline{z}$ the value $K_2 z^i_{RP}$ was used, where $K_2 < 1$. In this way, a lot of variables can be fixed early during the B&B-search and the speed of the algorithm increases.

8. Time Limit.

The intention of all previous modifications is to speed up the Branch-and-Bound algorithm. Unfortunately, still no guarantee can be given about the time needed for finding an optimal solution. Before the algorithm is applied in a CAT environment, this problem should be overcome in one or the other way. A rather straightforward way is to limit the time available for finding the optimal solution or to limit the number of solutions examined. Although the solution provided by this modification is less than optimal, the precision of the CAT procedure will probably not deteriorate.. When the B&B algorithm searches for the optimal solution, the best items are generally chosen early in this search.

Later on only small improvements are found. So, a lot of effort may be invested in finding only small improvements, that is in selecting items that only slightly improve the measurement precision. Based on this observation the time or number of solutions examined can be limited when the shadow test approach is used. After all, only the best item in the shadow test is presented to the candidate. So for choosing the best item limiting the search need not be always a problem.

Examples.

The modifications of the B&B algorithm presented in the previous section were applied in empirical CATs. In this way, the effects of the modifications could be examined. Both an admission test without item sets and a high-stakes reading test with item sets were used. For the admission test three modifications were of interest; branching strategy, node selection and the use of feasible starting solutions. Besides, priority ordering and the introduction of integer slack variables were examined for the reading test.

To run this study, CAT software developed at the University of Twente was used. This software made calls to the solver in the CPLEX 6.5 package (ILOG, 1999) to calculate the shadow tests. For the branching strategy (BS) five options were at hand in the CPLEX package:

1 branch on variable with minimum infeasibility,
2 branch variable automatically selected,
3 branch on variables with maximum infeasibility,
4 branch based on pseudo costs,
5 branch based on pseudo-reduced costs

The last two strategies are more intelligent branching rules. For node selection (NS) four options were at hand:

1 depth-first search,

2 best-bound search,

3 best-estimate search.

4 alternative best-estimate search

The seventh modification, dealing with near integer variables could not be implemented in the CPLEX package. All simulations were carried out on a Pentium II-266, 64 Mb computer.

Example 1.

The item pool for the admission test contained 753 items. The item pool fitted the three-parameter logistic model. So, the chance that candidate j obtains a correct answer to an item i was equal to

$$P_i(\theta_j) = c_i + (1 - c_i)\frac{\exp[1.7a_i(\theta_j - b_i)]}{1 + \exp[1.7a_i(\theta_j - b_i)]}, \qquad i = 1, ..., N \qquad j = 1, ..., M. \tag{12}$$

where θ_j is the ability level of candidate j, a_i the discrimination parameter, b_i the difficulty parameter and c_i the guessing parameter of item i. The general model for selecting the k-th item is presented in Equation 1 through Equation 2f. Since this item pool had no item sets no constraints at set level (Equation 2b), or constraints defining the number of items to be chosen per set (Equation 2e) needed to be formulated. The final model consisted of 753 variables and 25 constraints. The length of the CAT was set equal to 50 items.

For this problem, three modifications were relevant; problem selection, branching strategy and the use of feasible starting solutions. One hundred CATs were simulated for each combination of modifications. In this simulation study, the ability parameters of the examinees were standard normally distributed. The initial estimate of θ was set equal to $\widehat{\theta}_0 = 0$. The average time needed to select the next item was used as a measure to compare the settings. For problem selection and branching strategy, the results are presented in Table 1.

=================

Insert Table 1 at about here

=================

As can be seen, the best problem selection rule was best-estimate search, and the best braching strategy was the option of automatically selecting the branching variable.

Finally, for several combinations of problem selection and branching strategy, feasible starting solutions were used. The effects of a the starting solutions depended on the other modifications of the algorithm. In one case, a small improvement in performance was found. In the other, the use of feasible starting solutions resulted in a small deterioration of the speed. However, in general the use of feasible starting solutions did not result in a great improvement of performance.

Example 2.

The second pool consists of 723 items divided into 97 sets. The items fitted the three-parameter logistic model. The general model for selecting a next item is provided in Equation 1 through Equation 2f. In this application at set level 18 constraints were defined, at item level 22 constraints, and to ensure that the numbers of items selected per set were between their lower and upper bounds 97 additional constrains were needed. So, the number of constraints in this model was 139. For security reasons, the exact formulation of the model can not be given.

It should be noted that in a CAT with item sets two degrees of complexity occur. Since a stimulus can only be presented once, a minimum number of items has to be selected from the same set consecutively. When a stimulus is selected, but the minimum number of items is not presented jet, only the items that belong to the specific set have to be searched for the next item. When this minimum number is reached, the next item is chosen either from the set or from a new one. The latter problem is much harder to solve, since the number of items to be chosen from is much larger.

In order to illustrate the modifications, one hundred CATs were simulated for several combinations of modifications. The results are presented in Table 2. Each CAT consisted of 31 items. The ability parameters of the examinees were randomly distributed, and the initial ability estimates were set equal to $\widehat{\theta}_0 = 0$. The time needed for selecting the next item varied from 0.5 to 2.8 seconds.

==================

Insert Table 2 at about here

==================

In this example, the settings were optimized in the following manner. First the problem selection rule was modified. The depth-first search performed best. This is a remarkable result because in the depth-first strategy no information about the variables is taken into account. Then integer slack variables were added to the formulation. Adding these variables did not result in any improvement. Then, the best branching strategy was investigated. Branching on variables with maximum infeasibility performed best. Furthermore, integer slack variables were introduced again. But also with different branching strategies, introducing these variables did not improve performance in any way. Subsequently, a priority ordering was introduced. This modification improved the performance of the algorithm. Without integer slack variables the improvement was even larger. Finally the use of feasible starting solutions was studied. However, this modification did not result in a faster algorithm. In this example, the problems were also scanned for SOS's. Unfortunately no SOS's were present, so this modification could not be evaluated.

Discussion

In this paper, several modifications of the B&B algorithm have been studied. For the two examples at hand the following conclusions could be drawn. The use of feasible starting solution did not improve the performance of the algorithm. However, introducing feasible starting solutions guarantees that a solution to the problem of selecting the next item was found, even when the time available for selecting the next item is limited. In both examples, the next item was selected within a small amount of time. Therefore, no time limit was needed. On the other hand, when larger item pools or more constraints are involved in the test assembly process, the B&B search might take a lot of time and a time limit might be necessary for finding the next item in a reasonable amount of time. In those cases, the use of feasible starting solutions still is a good option. From this study it should be concluded that using a feasible starting solution is a conservative strategy. It depends on the test assemblers policy, whether it should or should not be used.

Introducing slack variables did not prove to be very useful in the second example. This might depend on the example. However, it is also possible that the solver is very efficient in handling the kind of constraints defined in Equation 8, while introducing additional variables deteriorates the performance. Introducing a priority ordering turned out to be an important modification. It resulted in an increased speed for all combinations of modifications investigated. Both for the branching strategy and for the problem selection strategy no general recommendations can be given. Both examples showed great differences. It should however be mentioned that choosing the best settings for these modifications resulted in a considerable improvement of performance. Since no SOS's occurred in the examples, their effect was not studied in practise.

From the results it can be concluded that the performance of the algorithm increases considerably when modifications are used. Even for large and complicated test assembly problems good solutions were found in a small amount of time. The modifications are effective in limiting the search. However, when item selection still takes too much time, the idea to limit the time available for the B&B search can be very useful.. An other idea is to carry out computations in advance. When a candidate is answering a question, meanwhile, the next item can be selected.

Besides these technical measures to improve performance of the B&B algorithm, current developments in Information Technology are on our side. For example, in this study the 6.5 version of CPLEX was used. Where in an earlier version, the optimal solution could not be obtained for some combinations of modifications, CPLEX 6.5 did not have these problems. Even the most time consuming calculation took less than thirty seconds.

References

J. J. Adema. Implementations of the branch-and-bound method for test construction problems. *Methodika*, 6:99–117, 1992.

J. J. Adema, E. Boekkooi-Timminga, and W. J. van der Linden. Achievement test construction using 0-1 linear programming. *European Journal of Operations Research*, 55:103–11, 1991.

R. D. Armstrong, D. H. Jones, and Z. Wang. Network optimization in constrained standardized test construction. *Applications of Management Science*, 8:189–212, 1995.

E. M. L. Beale and J. A. Tomlin. Special facilities in a general mathematical programming system for non-convex problems using ordered sets of variables. In J. Lawrence, editor, *Proceedings of the 5th International Conference on Operations Research*. Tavistock, London, 1969.

H. Crowder, E. L. Johnson, and M. Padberg. Solving large-scale zero-one linear programming problems. *Operations Research*, 31:803–834, 1983.

J. J. H. Forrest, J. P. H. Hirst, and J. A. Tomlin. Practical solution of large mixed integer programming problems with umpire. *Management Science*, 20:736–773, 1974.

I. ILOG. *CPLEX 6.5 [Computer Program and Manual]*. Incline Village, NV: ILOG, 1999.

G. G. Kingsbury and A. R. Zara. Procedures for selecting items for computerized adaptive tests. *Applied Measurement in Education*, 2:359–375, 1991.

G. Mitra. Investigation of som branch and bound strategies for the solution of mixed integer linear programs. *Mathematical Programming*, 4:155–170, 1973.

G. L. Nemhauser and L. A. Wolsey. *Integer and combinatorial optimization*. New York, NY: Wiley, 1988.

D. O. Segall, K. E. Moreno, and R. D. Hetter. Item pool development and evaluation. In W. A. Sands, B. K. Waters, and J. R. McBride, editors, *Computerized Adaptive Testing: From Inquiry to Operation*, pages 117–130. American Psychological Association, Washington, DC, 1997.

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

E. Timminga, W. J. van der Linden, and D. A. Schweizer. *ConTEST 2.0: A decision support system for item banking and optimal test assembly (computer program and manual)*. Groningen, The Netherlands: iec ProGAMMA, 1996.

W. J. van der Linden. Constrained adaptive testing with shadow tests. In W. J. van der Linden and C. A. W. Glas, editors, *Computerized Adaptive Testing: Theory and Practice*, pages 27–53. Kluwer Academic Pubishers, Boston, MA, 2000.

W. J. van der Linden and L. M. Reese. A model for optimal constrained adaptive testing. *Applied Psychological Measurement*, 22:259–270, 1998.

H. P. Williams. *Model Building in Mathematical Programming*. Chichester, England: Wiley, 1990.

W. L. Winston. *Operations Research, Applications and Algorithms*. (3rd ed.). Belmont, CA: Wadsworth Publishing Company, 1994.

Table 1

Time(s) for selecting the next item.

Problem	Branching Strategy					
Selection	1	2	3	4	5	Average
1	7.40	6.46	7.84	6.37	6.85	6.98
2	7.51	5.26	7.71	5.82	6.23	6.51
3	7.40	5.24	6.95	5.58	6.11	6.26
4	7.76	5.83	7.44	5.85	6.63	6.70
Average	7.52	5.70	7.48	5.91	6.46	

Table 2

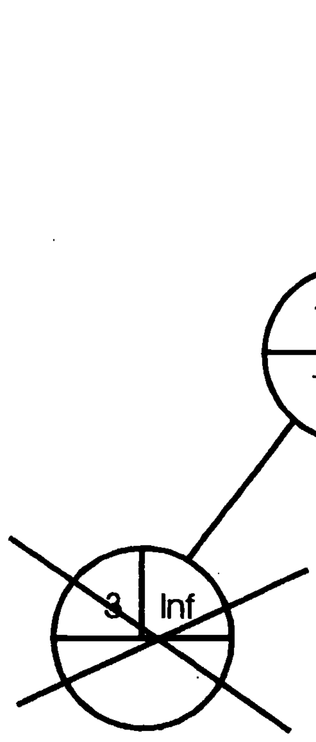
Optimal settings B&B algorithm for a Reading Comprehension test.

Modifications					Average time(s) for selecting the next item
SS	IS	PO	BS	PS	
			2	1	1.56
			2	2	1.65
			2	3	1.68
			2	4	1.75
	u		2	1	2.81
	u		2	2	2.13
	u		2	3	2.54
	u		2	4	2.56
			1	3	1.40
			3	3	1.05
			4	3	1.73
	u		1	3	2.52
	u		3	3	1.47
	u		4	3	2.14
	u	u	1	3	1.48
	u	u	3	3	1.28
	u	u	4	3	1.42
		u	1	3	1.27
		u	3	3	1.13
		u	4	3	0.50
u	u		1	3	1.73
u	u		4	3	2.36
u	u	u	1	3	1.47
u	u	u	4	3	1.50

u - used.

Figure Caption

Figure 1 : Tree representing Branch-And-Bound Search


0
1,9
71/80 ,1,0
1
1,8
1, 10/85 ,0
2
1,2
0,1,0
3
Inf
4
1,6
1,0,1

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

RR-98-10	W.J. van der Linden, *A Discussion of Some Methodological Issues in International Assessments*
RR-98-09	B.P. Veldkamp, *Multiple Objective Test Assembly Problems*
RR-98-08	B.P. Veldkamp, *Multidimensional Test Assembly Based on Lagrangian Relaxation Techniques*
RR-98-07	W.J. van der Linden & C.A.W. Glas, *Capitalization on Item Calibration Error in Adaptive Testing*
RR-98-06	W.J. van der Linden, D.J. Scrams & D.L.Schnipke, *Using Response-Time Constraints in Item Selection to Control for Differential Speededness in Computerized Adaptive Testing*
RR-98-05	W.J. van der Linden, *Optimal Assembly of Educational and Psychological Tests, with a Bibliography*
RR-98-04	C.A.W. Glas, *Modification Indices for the 2-PL and the Nominal Response Model*
RR-98-03	C.A.W. Glas, *Quality Control of On-line Calibration in Computerized Assessment*
RR-98-02	R.R. Meijer & E.M.L.A. van Krimpen-Stoop, *Simulating the Null Distribution of Person-Fit Statistics for Conventional and Adaptive Tests*
RR-98-01	C.A.W. Glas, R.R. Meijer, E.M.L.A. van Krimpen-Stoop, *Statistical Tests for Person Misfit in Computerized Adaptive Testing*
RR-97-07	H.J. Vos, *A Minimax Sequential Procedure in the Context of Computerized Adaptive Mastery Testing*
RR-97-06	H.J. Vos, *Applications of Bayesian Decision Theory to Sequential Mastery Testing*
RR-97-05	W.J. van der Linden & Richard M. Luecht, *Observed-Score Equating as a Test Assembly Problem*
RR-97-04	W.J. van der Linden & J.J. Adema, *Simultaneous Assembly of Multiple Test Forms*
RR-97-03	W.J. van der Linden, *Multidimensional Adaptive Testing with a Minimum Error-Variance Criterion*
RR-97-02	W.J. van der Linden, *A Procedure for Empirical Initialization of Adaptive Testing Algorithms*
RR-97-01	W.J. van der Linden & Lynda M. Reese, *A Model for Optimal Constrained Adaptive Testing*

...

<u>Research Reports</u> 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