Report Nr 05

A new heuristic to solve the item selection problem: outline and numerical experiments

A.J.R.M. Gademann

Cito

A NEW HEURISTIC TO SOLVE THE ITEM SELECTION PROBLEM: OUTLINE AND NUMERICAL EXPERIMENTS

by

A.J.R.M. Gademann

Report Nr 05

Project 'Optimal Item Selection'

CITO

Arnhem, the Netherlands (1989)

General Introduction

The purpose of Project 'Optimal Item Selection' is to solve a number of issues in automated test design, making extensive use of optimization techniques. To this end, there has been a close cooperation between the project and, among others, the department of Operations Research at Twente University. In each report, one or several theoretical issues are raised and an attempt is made to solve them. Furthermore, each report is accompanied by one or more computer programs, which are the implementations of the methods that have been investigated. In due time, requests for these programs can be sent to the project director.

T.J.J.M. Theunissen,
project director.

Contents

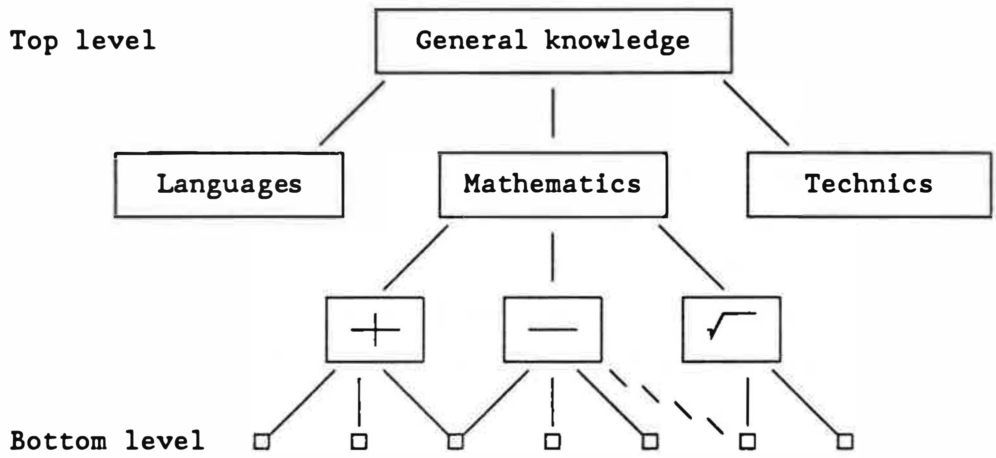
Introduction

In this report a method is presented to find a solution to a simplified version of the item selection problem. Based on a special relaxation and an aggregation method, a heuristic is developed that solves the problem fast and accurately. Test results will be presented and compared with the results of several (quasi) exact and heuristic solution methods. Some concluding remarks will be given on the algorithm and the test results. The Rasch model is the item response model that is the basis of the mathematical model.

1 Specification of the item selection problem and an LP-relaxation

The item selection problem is the problem of selecting a number of items from an item bank (a set of available items), such that the resulting test satisfies several conditions. An item in the item bank is characterized by its difficulty parameter - recall that we use the Rasch model - and the categories the item belongs to. The item bank can be represented by a layered network (see figure 1).

(1.1) Example of a network representation of an item bank



Every level of the network is a specification of the higher level, and the bottom level represents the individual items. In general we can specify conditions on each level of the network. For example:

- individual items (bottom level): IF item i in test THEN NOT item j;
- categories (subsets of items): the number of items selected from category k is at least a fraction f_k of the total number of items in the test;
- item bank (top level): the test information function must exceed the target information function.

Other restrictions we can think of are for example a maximum on the total duration of the test or a maximum number of items in the test. Besides there are several objectives possible.

In this report we will study a simple version of the item selection problem. The objective is to minimize the number of items in a test while reaching a target information for several ability levels. This problem, which has been studied by several authors ([4], [5], [6], [9], [11]), can be represented by the following model:

$$\text{(1.2)} \quad \text{IP} \quad \min f(x), \quad \text{such that}$$
$$Ax \geq b$$
$$x_j \in \{0,1\},\ j = 1,..,n$$

where f(x), b and A are specified as follows:

$$\text{(1.3)} \quad f(x) = \sum_{j=1}^{n} x_j$$
$$b_i = T(\theta_i),\ i = 1,..,m$$
$$= \text{the target information for ability level } \theta_i,$$
$$a_{ij} = I(\theta_i,j),\ i = 1,..,m,\ j = 1,..,n$$
$$= \text{the item information of item } j \text{ for ability level } \theta_i$$

based on the Rasch-model (e.g. see [7])

The interpretation of the decision variables x_j is as follows. With every item j in the item bank corresponds a (0,1) decision variable x_j: $x_j=0$ means that item j is not selected in the test, $x_j=1$ means that item j is selected.

A relaxation of problem IP that has been extensively studied is:

$$\text{(1.4)} \quad \text{LP} \quad \min \sum_{j=1}^{n} x_j \quad \text{such that}$$
$$Ax \geq b$$
$$0 \leq x_j \leq 1,\ j = 1,..,n$$

LP is a linear programming problem with just a few constraints. Therefore we can obtain a feasible solution to IP by solving LP with for example the simplex method and use a rounding method. But apart from the fact that this solution is not always optimal, the computation time is high in comparison with heuristics. A solution of LP can eventually be used as a starting-point for a branch-and-bound procedure to produce a solution to IP.

When we look at the tests that result from applying (quasi) exact methods to solve IP (for example by first solving LP), we observe that the items are always selected in one or more groups of items with about the same difficulty level of the items in one group. Therefore we refer to those groups by the term clusters, since the selected items cluster around several difficulty parameter values. The number of clusters can differ from 1 to at most m (the number of target information points, that is the number of constraints), depending on the item bank and the target information points.

Razoux Schultz ([9]) developed a heuristic that is based on this observation. In that heuristic two cluster points (i.e. difficulty parameters) are calculated using a special function. These cluster points are such that if a test satisfying the target information restrictions is constructed from an item bank which is assumed to contain an infinite number of items at each difficulty level (an ideal item bank, that of course does not exist in practice) and if this test consists of at most two clusters, then all selected items will have difficulty parameter values equal to one of the calculated cluster points. If in a real test construction problem the condition of having at most two cluster points using an ideal bank is satisfied, then the optimal solution of such a problem using the real item bank will be found by selecting only items with difficulty parameter equal to or near one of the calculated cluster points. The heuristic of Razoux Schultz selects items pairwise with difficulty level around the calculated cluster points until a feasible solution is obtained.

The following complications may occur when this heuristic is applied:

- the theoretical number of clusters (using a ideal item bank) is larger than 2. In this case the solution will in general not be optimal.

- The target information function is not symmetrical. In that case the items indeed have to be selected near the cluster points but not pair wise, i.e. the number of items in the two clusters can differ significantly. This will not be detected by the heuristic since no information about the relative or absolute number of items in each of the clusters is available and the items are always selected pair wise.

- The number of available items with difficulty level near a cluster point differs considerably for the two cluster points. In that case the information of the items selected near the cluster points is not the same for both points, so the number of items that have to be selected can also differ considerably for the two clusters. This is also not detected for the same reasons as mentioned above.

- obviously a combination of the foregoing situations might cause complications as well.

In the next section another relaxation of the integer programming problem IP is introduced, which will be shown to have some useful properties.

2 A linear relaxation with useful properties

A relaxation of IP that will appear to be very useful is given by:

(2.1) RP $\quad \min f(x) \quad$ such that

$$\tilde{A}x \geq b$$

$$x_j \geq 0,\ j = 1,..,n$$

$\tilde{A}$ is the coefficient matrix of an item bank with one item at every possible difficulty level in the range [-3.0, 3.0]. Since there are infinitely many levels in that range this is only a theoretical assumption. However we could approximate such an item bank by an item bank that contains one item of every difficulty level from -3.00 to +3.00 with an increment of α. So difficulty levels -3.00, $-3.00+\alpha$, $-3.00+2\alpha,\ldots,3.00$ are present. It is clear that the approximation gets better when α gets smaller. We denote the primal solution to RP by x^* and the associated dual solution by π^*. We may remark that problem RP is the only problem in this report that uses (approximately) ideal data. The other problems that are specified use the real data of the item selection problem that has to be solved.

What is the interpretation of the relaxation RP? Well, since a decision variable can assume any non-negative value, available items can be selected more than once. Therefore problem RP is the same as the LP relaxation (1.4) for an ideal bank, where in RP all items with the same difficulty parameter are represented by just one decision variable x_j. And since an ideal item bank contains items at all difficulty levels, the optimal solution to RP will consist of only items with difficulty level equal to a cluster point. So the solution x^* provides us with the cluster points, namely the difficulty parameters d_j of the items j with $x_j^* > 0$. If we define x^I for example as follows:

(2.2) $\quad x^I := \lceil x^* \rceil \quad$ (i.e. x^* rounded up component wise)

we also obtain an indication of the number of items in the various clusters, namely approximately $x^I{}_j$ items in the cluster around d_j. (A different rounding method to define x^I can be used but that is not really relevant at this moment.) Another interesting property is that we can easily derive a lower bound Z_{lb} to IP from x^*:

(2.3) $$Z_{lb} = \left\lceil \sum_{j=1}^{n} x_j^* \right\rceil$$

We could use the obtained information about the clusters to adapt the cluster point method of Razoux Schultz. A minor problem remains if we do so. For we know (approximately) the number of items to be selected in the clusters, but we do not know which items have to be selected. We need a decision rule to decide which of the items with item difficulty parameter near but not equal to a cluster point, should be selected.

Now for a moment we shall let these results rest and investigate in what way we can use π^*, the optimal dual solution to RP. For that purpose we specify the following problem:

(2.4) AP $$\min \sum_{j=1}^{n} x_j \quad \text{such that}$$
$$\pi^* Ax \geq \pi^* b$$
$$x \in \{0,1\}$$

Problem AP is derived from IP by aggregating the constraints using weights π_i^*. We recall that only problem RP uses ideal data, so the matrix A in (2.4) is again the matrix with real data. AP is closely related to the well-known knapsack problem and can as such be easily solved by sorting the items j such that:

(2.5) $$(\pi^* A)_1 \geq \ldots\ldots \geq (\pi^* A)_n$$

and subsequently setting x_j to:

(2.6) $x_j = 1$, $j = 1,\ldots,r$,
$= 0$, $j = (r+1),\ldots,n$, with:

$$\sum_{j=1}^{r} (\pi^*A)_j \geq \pi^*b,$$

$$\sum_{j=1}^{r-1} (\pi^*A)_j < \pi^*b$$

The following theorem gives a useful property of the solution x of AP.

(2.7) Theorem: if the solution x of AP satisfies: $Ax \geq b$, then x is optimal for IP.

Proof: suppose x is not optimal for IP.
Then there exists a feasible solution y for IP with $f(y)<f(x)$. But then the following implications hold:
y is feasible for IP $\Rightarrow$ $Ay \geq b$ $\Rightarrow$ $\pi^*Ay \geq \pi^*b$ $\Rightarrow$
y is feasible for AP $\Rightarrow$ x is not optimal for AP,
which leads to a contradiction $\square$

So theorem (2.7) tells us that a solution to AP is also optimal for IP if it is a feasible solution for IP. If the solution is not feasible for IP we have to adjust the weights π_i^*. It is obvious that the restrictions that are not satisfied, should get a relatively higher weight and that the restrictions that have slack should get a relatively lower weight.

If we define x^k as the solution found after k adjustments of the weights and π^k as the vector of corresponding weights, the iterative adjustment of the weights can be given by:

(2.8) $y^k := Ax^k - b$
$\pi_i^{k+1} := \max \{ 0 , \pi_i^k + t^k * y_i^k \}$

The problem is how to choose a proper value for t^k.

The adjustment (2.8) is often used in so-called subgradient optimization where a Lagrange relaxation of IP including weights π_i^k is applied instead of problem AP. This Lagrange relaxation is given by:

$$(2.9) \quad \text{LGR} \quad \min L(x^k) = \sum_{j=1}^{n} \left[1 - \sum_{i=1}^{m} \pi_i^k * a_{ij} \right] * x_j^k + \pi^k * b$$

$$= \sum_{j=1}^{n} C_j * x_j^k + \pi^k * b \, , \qquad \text{such that}$$

$$x_j^k \in \{0,1\}, \quad j = 1,..,n$$

The solution to LGR is given by:

$$(2.10) \qquad x_j^k = \begin{cases} 1 & \text{if} \quad C_j < 0 \\ 0/1 & \text{if} \quad C_j = 0 \\ 0 & \text{if} \quad C_j > 0 \end{cases}$$

The solution process using LGR and adjustments 2.8 is called subgradient optimization because y^k is a subgradient of $L(x)$ in x^k. It can be shown ([8]) that $\{L(x^k)\}$ converges to the optimal value of $f(x)$ if the factors $\{t^k\}$ satisfy the following conditions:

$$(2.11) \quad t^k \rightarrow 0 \quad \text{for} \quad k \rightarrow \infty,$$

$$\sum_{k=1}^{\infty} t^k = \infty$$

A choice for t^k that satisfies 2.11 and that is often successful, is:

$$(2.12) \quad t^k = f * \frac{Z_{ub} - Z_{lb}}{|| y^k ||} \, , \text{ with: } || y^k || = \sum_{i=1}^{m} (y_i^k)^2$$

Subgradient optimization according to (2.8) .. (2.12) was implemented by Kester ([6]), following (exactly) an article by Beasley ([3]). For the factor f in (2.12) an initial value of 2.0 was chosen and f was halved whenever there was no substantial improvement in the solution. The algorithm was halted when $f < 0.008$. The algorithm happened to converge very slowly due to a zig-zag motion that did not damp very well. This can be explained if we examine the results of a trial-and-error process to adjust the weights π_i^k. For it appeared that the optimal weights π_i^{∞} differed often only significantly from the start weights π_i^* in the fourth or fifth decimal and that the solution was always very sensitive for changes in these decimals, which implies an adjustment using (2.12) with $f < 0.001$. Therefore starting the iterative process with $f = 2.0$ often first leads to a detoriation instead of convergence to the optimal value π^{∞}, and stopping when $f < 0.008$ is too soon.

The fact that π^{*} and π^{∞} differ only slightly means that sorting the items such that $(\pi A)_1 \geq .. \geq (\pi A)_n$ will lead to more or less the same sorted sets for $\pi = \pi^{*}$ and for $\pi = \pi^{\infty}$. So if we sort the items using π^{*}, the optimal items will be "somewhere in front of the sorted list of items".

A final remark we make is that problems AP and LGR do not differ essentially since for both problems the solution consists of those items j for which $(\pi A)_j$ is the largest. The only difference is that the solution of AP can also contain items for which $(\pi A)_j < 1$.

In the next section a heuristic will be presented that is based on the results and observations from the various problems that are discussed so far, namely IP, LP, RP, AP and LGR.

3 A heuristic to solve the item selection problem

The heuristic that will be presented in this section is mainly based on the following results and observations made so far:

- from the primal solution x^* of RP, using (approximately) ideal data we know how many clusters occur in the solution to IP, and around which difficulty parameters the items cluster. Besides we know approximately the number of items in each cluster, given by x^I.

- using the dual solution π^* of RP we can order the items such that $(\pi^*A)_1 \geq .. \geq (\pi^*A)_n$. If the solution to AP is feasible for IP, the optimal solution to IP is found. Otherwise the optimal items will be somewhere in front of the sorted list of items.

The resulting heuristic is given by:

(3.1) Heuristic to solve the item selection problem IP

<u>Step 1</u> Solve RP. Solution: x^*(primal), π^*(dual).
$x^I := \lceil x^* \rceil$

<u>Step 2</u> Sort items such that $(\pi^*A)_1 \geq .. \geq (\pi^*A)_n$

<u>Step 3</u>
```
j:=0;
x_j := 0, j = 1,..,n;(no items selected)
REPEAT j:=j+1;
       IF cluster(item j) IS NOT full
       THEN x_j := 1 (select item j)
UNTIL all clusters full (x^I_j items in cluster j for all j)
      OR j = n (not enough items available to fill all
               clusters);
```

<u>Step 4</u>
```
IF x feasible
THEN go to step 5
ELSE REPEAT select item "greedily" (f.e. as described below)
     UNTIL (x feasible) OR (all items selected)
```

Step 5 IF x feasible
THEN backtrack (check whether selected item(s) can be removed without violating a restriction of IP)
ELSE there is no solution for IP

There are several versions possible for the rounding procedure to determine x^I in step 1. Another possibility is e.g.:

(3.2) $x_j^I :=$ ROUND(x_j^*), $j = 1,..,n$.

The "greedy" selection of items in step 4 can be implemented in various ways. The method used in the testing of the algorithm is selecting those items that give the largest contribution to one of the violated restrictions, so the selected item $j^* \in \{1,2,..,n\}$ maximizes:

(3.3) $\max\limits_i \dfrac{a_{ij}}{b_i - A^i x}$, $i = 1,..,m$

In the next section some test results will be presented.

4 Results

In the numerical experiments we used 24 simulated item banks. The difficulty parameters of the items in the bank were sampled from 8 different distributions, where only sampled parameters with values in the interval [-3.2 , +3.2] were accepted. The 8 distributions used were:

(4.1)

Bank nr.	Distribution
1	Uniform(-3,+3)
2	N(0,1)
3	N(0,2)
4	N(0,4)
5	N(-2,1)
6	N(-2,2)
7	N(-2,4)
8	N(10,25)

With each of these distributions three item banks were simulated with respectively 300, 500 and 1000 items, which resulted in the 24 item banks (8 x 3) mentioned before.

There were 4 test problems specified. The specifications are given in table (4.2), as well as the solution to the associated problems RP.

(4.2) Specifications of the four test problems

Problem no.	Targets		Continuous	Difficulty	CPU time
	θ	$I(\theta)$	solution	parameter	(in sec.)
1	-2 0 2	6 11 6	25.83 25.81	-0.81 0.81	3.80
2	-2 0 2	5 5 10	16.91 38.85	-1.84 1.84	2.90
3	-2 0 2	10 5 5	38.85 16.91	-1.84 1.84	2.90
4	-2 0 2	7 7 7	26.11 26.14	-1.66 1.67	8.80

We notice that all test problems have theoretically two cluster points. There are both symmetric and asymmetric target information functions. The heuristic (3.1) was applied to solve the four test problems for all 24 item banks. So a total number of 96 problems is solved. The results of these numerical experiments are presented in tables (4.3) to (4.7) together with some statistics. In these tables the results are compared with the results of five other solution methods, namely:

- three other heuristics: Mindev ([4], [9]), Clusterpoints([9]) and Subgradient([6]);
- a quasi-exact method: Multi (Simplex for LP + rounding as in [5]);
- an exact algorithm: the Land and Doig algorithm for IP (see e.g. [10])

(4.3) Results for problem specification 1

Method Bank #items		Mindev	Subgr	Clust	Land	Multi	Heur3
1	300	53	53	52	52	52	53
	500	54	54	53	52	52	52
	1000	54	54	52	52	52	52
2	300	53	53	52	52	52	52
	500	53	53	52	52	52	52
	1000	54	54	52	52	52	52
3	300	54	54	52	52	52	53
	500	54	54	52	52	52	52
	1000	54	54	53	52	52	52
4	300	53	53	53	52	52	52
	500	54	54	52	52	52	52
	1000	54	54	52	52	52	52
5	300	98	98	-	98	98	98
	500	84	84	84	84	84	84
	1000	55	55	60	53	54	54
6	300	54	54	56	53	53	53
	500	53	53	53	52	53	53
	1000	54	54	52	52	52	52
7	300	53	53	52	52	52	53
	500	54	54	53	52	52	53
	1000	53	53	52	52	52	52
8	300	54	54	53	52	52	53
	500	53	53	53	52	52	52
	1000	54	54	52	52	52	52
Optimal		2	2	14	24	22	17
+1		9	9	7		2	7
+2		13	13				
> +2				2			
No solution				1			
Rank 1		2	2	14	24	22	17
2		1	1	1		2	2
3				2			2
4		6	6	4			3
5		15	15				
6				3			

(4.4) Results for problem specification 2

Method Bank	 #items	Mindev	Subgr	Clust	Land	Multi	Heur3
1	300	57	57	62	57	57	57
	500	57	56	63	56	56	56
	1000	57	57	62	56	56	56
2	300	58	58	63	58	59	58
	500	58	58	63	57	58	58
	1000	57	57	63	56	57	57
3	300	57	57	63	57	57	57
	500	57	56	62	56	57	57
	1000	57	56	62	56	56	56
4	300	57	57	62	57	57	57
	500	57	57	63	56	57	57
	1000	57	56	62	56	56	56
5	300	86	72	98	72	72	72
	500	80	68	93	67	67	67
	1000	62	62	62	61	62	62
6	300	62	60	61	60	60	61
	500	58	57	62	57	57	58
	1000	58	57	63	57	57	57
7	300	57	57	63	57	57	57
	500	57	57	63	57	57	57
	1000	57	57	62	56	57	57
8	300	58	58	63	58	59	58
	500	58	58	63	58	58	58
	1000	57	57	63	56	57	57
Optimal		8	16		24	15	15
+1		13	8	2		9	9
+2		1					
> +2		2		22			
No solution							
Rank 1		8	16		24	15	15
2		6	6	1		6	6
3		1				1	1
4		2	2	1			2
5		6				2	
6		1		22			

(4.5) Results for problem specification 3

Method Bank	 #items	Mindev	Subgr	Clust	Land	Multi	Heur3
1	300	57	57	63	57	57	57
	500	57	56	62	56	56	56
	1000	57	56	62	56	56	56
2	300	59	59	65	58	59	59
	500	58	57	63	57	57	57
	1000	57	57	63	57	57	57
3	300	58	57	63	57	58	57
	500	57	57	63	57	57	57
	1000	57	56	62	56	56	56
4	300	57	57	63	57	57	57
	500	57	57	63	57	57	57
	1000	57	56	63	56	56	56
5	300	xx	xx	xx	xx	xx	xx
X	500	200	nn	nn	200	nn	200
	1000	92	92	96	92	92	93
6	300	118	118	118	118	118	118
	500	65	65	74	64	65	64
	1000	63	62	72	62	62	62
7	300	61	61	69	61	61	61
	500	59	59	66	59	59	59
	1000	57	57	63	57	58	58
8	300	57	57	63	57	57	57
	500	57	57	62	56	57	57
	1000	57	56	62	56	56	56
Optimal		12	20	2	23	18	19
+1		11	3			5	4
+2							
> +2				21			
No solution							
Rank 1		12	20	2	23	18	19
2		2	2			2	2
3		1	1			1	
4		1				2	1
5		7					1
6				21			

X = not included in statistics
nn = not known, no results available
xx = infeasible

(4.6) Results for problem specification 4

Method Bank	 #items	Mindev	Subgr	Clust	Land	Multi	Heur3
1	300	55	53	53	53	54	53
	500	55	54	53	53	53	53
	1000	55	54	53	53	54	54
2	300	55	54	54	54	54	54
	500	55	54	53	53	54	54
	1000	55	55	53	53	53	53
3	300	55	54	53	53	53	53
	500	55	54	53	53	54	53
	1000	55	55	53	53	54	53
4	300	55	54	53	53	53	53
	500	55	54	53	53	53	53
	1000	55	54	53	53	54	53
5	300	136	135	--	135	135	135
	500	108	107	--	107	107	107
	1000	64	63	89	63	63	63
6	300	66	62	86	62	62	63
	500	56	56	61	55	56	56
	1000	56	55	61	55	56	55
7	300	55	54	57	54	54	55
	500	55	54	54	54	54	54
	1000	55	53	53	53	53	53
8	300	55	53	55	53	53	53
	500	55	53	53	53	53	53
	1000	55	54	53	53	54	53
Optimal			12	16	24	15	19
+1		8	10			9	5
+2		15	2	1			
> +2		1		5			
No solution				2			
Rank 1			12	16	24	15	19
2		1	1			1	1
3			2			2	2
4		1	3			5	2
5		7	6	1		1	
6		15		7			

In tables (4.3) to (4.6) we see that especially for item banks of type 5 the solutions are extreme and depend strongly on the number of items in the bank. This is easily explained when we notice that the distribution of the item difficulty parameters for these item banks is the worst in the sense that only a few more difficult items are available as compared with the number of easier items (see (4.1)).

(4.7) Overall statistics for the four problem specifications

Method		Mindev	Subgr	Clust	Land	Multi	Heur3
Optimal		22	50	32	95	70	70
+1		41	30	9		25	25
+2		29	15	1			
> +2		3		50			
No solution				3			
Rank 1		22	50	32	95	70	70
2		10	10	2		11	11
3		2	3	2		4	5
4		10	11	5		7	8
5		35	21	1		3	1
6		16		53			
Minimal CPU time *	3	0.4	33	0.5	4	6	1.0
	5	0.6	48	0.5	6	11	1.6
	1	0.7	103	0.5	82	76	3.2
Mean CPU time *	3	0.7	59	0.7		33	4.0
	5	0.7	70	0.7		72	4.0
	1	1.0	134	0.7		168	6.2
Maximal CPU time *	3	2.3	323	1.2	**	57	25.1
	5	1.6	201	1.1	**	123	24.9
	1	1.6	169	1.3	**	272	23.3

Heur3 was executed on an Olivetti XT (8 Mhz) with a numeric co-processor, the other algorithms where executed on a Victor AT (8 Mhz), also with a numeric co-processor. This AT was about two to three times faster than the XT.

* = in seconds, without CPU time for input and output, and without CPU time for solving RP for heur3 , because RP had to be solved only once for each of the four problem specifications (see (4.2)).

3 / 5 / 1 = item bank of 300 / 500 / 1000 items

** = more than 3 hours, therefore computing mean CPU time does not make sense.

It has to be noted that except for Heur3 the algorithms were applied to item banks that were sorted in order of increasing item difficulty parameter values.

From table (4.7) we see that for Heur3 the mean CPU times for item banks of 300, 500 or 1000 items do not differ a lot. This is due to the solutions with extreme CPU times (about Max. CPU). For these problems, most of the CPU time is spent to perform steps 4 and 5 of the algorithm, that is for those problems for which the solution to problem IP is much greater that the solution to problem RP, since in that case a lot of items are selected in step 4. The mean CPU times for the problems for which the solution to IP is about the same as the solution to RP, let's say (solution IP - solution RP) $\leq$ 3), is about:

(4.8)

Number of items in the bank	Mean CPU time (in sec.)
300	1.3
500	2.2
1000	4.6

From (4.8) we see that the CPU time is almost linear in the number of items in the bank, as long as steps 4 and 5 are not significantly involved (CPU time for STEP 1 not included).

For the algorithms Mindev and Clusterpoints we see that the CPU times are small and almost independent of the number of items in the item bank. For these algorithms the CPU time depends mostly on the number of items that have to be selected. It has to be noted that the algorithm Subgradient has to start with a feasible solution, and that this feasible solution was derived by Mindev. That explains the conformity of results for Mindev and Subgradient for example in table (4.3).

From the overall results in table (4.7) we may conclude that the heuristic Heur3 performs very well. Of course the exact Land and Doig algorithm is the most accurate, but due to its CPU times it has no practical value. Of course are the results of this algorithm of theoretical importance to check the accuracy and reliability of the other algorithms. Among the other algorithms Heur3 is the most accurate one, and besides it is fast and reliable, since there were no solutions that were "far from optimal". From this we can denote Heur3 as the best algorithm for practical use among the tested algorithms.

5 Concluding remarks

In this report a new heuristic is discussed to solve a simple version of the item selection problem. The heuristic is based on the observation of clustering of items in optimal solutions to item selection problems. Several operational research concepts have been used to obtain information about the solution and to construct the heuristic. The heuristic has been compared with several other solution methods. From this comparison we may conclude that the new heuristic is preferable above the other methods for practical use for this version of the problem, both because of its accuracy and its reliability. Some parts of the heuristic, that are needed especially when the demands (f.e. target information) and the supply (available items) do not fit very well, may be improved. Also a formalization of when and why the heuristic performs well is desirable but not yet accomplished.

Summary

In this report a simple version of the item selection problem, namely minimizing the number of selected items while satisfying some test information criteria (target information), is studied. A new heuristic, based on the observation of clustering of items in optimal solutions to item selection problems, is presented. The heuristic is compared with several other solution methods by solving 96 simulated test construction problems. Several operational research concepts are used to construct the heuristic.

References

[1] Balas E., "Cutting planes from conditional bounds: a new approach to set covering", Mathematical Programming 12(1980), pp. 19-36.

[2] Balas E. and A. Ho, "Set covering algorithms using cutting planes, heuristics, and subgradient optimization: a computational study", Mathematical Programming Study 12(1980), pp. 37-60.

[3] Beasley J.E., "An algorithm for set covering problem", European Journal of Operational Research 31(1987), pp. 85-93.

[4] Boomsma Y., *Item selection by mathematical programming*, Bulletin no. 47, Cito, Arnhem, 1986.

[5] Gademann A.J.R.M., *Item selection using multiobjective programming*, OIS report no. 01, Cito, Arnhem, 1987.

[6] Kester J.G., *Various mathematical programming approaches toward item selection*, OIS report no. 03, Cito, Arnhem, 1988.

[7] Lord F.M., *Applications of item response theory to practical testing problems*, Lawrence Erlbaum, Hillsdale, New Jersey, 1980.

[8] Poljak B.T., "A general method of solving extremum problems", Soviet Mathematics Doklady 8(1967), pp. 593-597.

[9] Razoux Schultz A.F., *Item selection using heuristics*, OIS report no. 02, Cito, Arnhem, 1987.

[10] Salkin H.M., *Integer programming*, Addison-Wesley, Reading, Massachusetts, 1975.

[11] Theunissen T.J.J.M., "Binary programming and test design", Psychometrika 50, pp. 411-420.

Titles of other research reports:

Nr. 01 Item selection using Multiobjective Programming.
A.J.R.M. Gademan (1987).

Nr. 02 Item selection using Heuristics.
A.F. Razoux Schultz (1987).

Nr. 03 Various mathematical programming approaches toward item selction.
J.G. Kester (1988).

Nr. 04 Minimizing the number of observations: a generalization of the Spearman-Brown formula.
P.F. Sanders, T.J.J.M. Theunissen and S.M. Baas (1988).