

## Automated Test Assembly Heuristics: Algorithms for the Rest of Us

Kirk A. Becker, Ph.D.  
James S. Masters  
Pearson VUE

*This paper will focus on the operational side of automated test assembly, with particular attention paid to characteristics of smaller programs. Issues such as ease of implementation, defining processes used by content developers, implicit rules and constraints, and algorithms for optimizing pool use will be addressed. This paper will focus on the Weighted Deviations Model (WDM) (Swanson & Stocking, 1993) heuristic, in combination with rules and logic geared to address specific program needs. Considerations such as small/shallow item banks, enemy items, efficient use of item banks, diverse representation within content areas, primacy of content specifications, and ease of implementation are addressed.*

### **Introduction:**

Automated test assembly (ATA) is a method for producing tests of a fixed length that satisfy a number of constraints or conditions (Parshall, Spray, Kalohn, & Davey, 2001). There are four primary approaches to ATA: heuristic-based test assembly, 0-1 linear programming (0-1 LP), network flow programming, and an optimal design approach. The heuristic-based approaches do not require modeling, and they are generally quicker than the other methods. Advantages of the 0-1 LP approach are that the heuristics will find the optimal solution, it is not possible to violate any constraints, and it is flexible. The power of the network flow programming approach is its speed (summarized from van der Linden, 1998). Finally, optimal design provides an approach to specifying the optimal distribution of item parameters, then used as input for other ATA methods.

With particular attention paid to characteristics of smaller programs, this paper will focus on the operational side of automated test assembly, addressing issues such as ease of implementation, defining processes used by content developers, implicit rules and constraints, and algorithms for optimizing pool use.

### **Theoretical Framework:**

"The weighted deviations model (WDM) heuristic is particularly promising for automated test assembly (ATA) because it is computationally straightforward and produces tests with desired properties under realistic testing conditions" (Lin, 2008). In situations where the item pool cannot fulfill all of the target-test constraints, the WDM heuristic will produce the best possible test, an advantage over 0-1 LP models (Lin, 2008). "Heuristics like the normalized weighted absolute deviation heuristic (NWADH) have gained popularity in a variety of disciplines because of their capability to produce near-optimal solutions to complex, difficult, or time-consuming optimization problems" (Luecht, 1998).

While there are many options for selecting items in automated test assembly, many of the more important decisions on the process lie outside of the specific algorithm used to select items. This paper will focus on the process of implementing the Weighted Deviations Model (WDM) (Swanson & Stocking, 1993) heuristic for a testing program, and on the combination of rules and logic needed to address and operationalize the algorithm. While some of these are specific to the WDM, many should apply to ATA in general.

The WDM is a method for quantifying the desirability of each available item in an item pool relative to the definition of the test being built. The test definition specifies a set of constraints (e.g., content specifications, psychometric characteristics, etc.). Tests are built sequentially, and the desirability of each item is always relative to the partially completed test and to the remaining items in the pool. Mathematically, the desirability of an item is defined as:

$$S_t = \sum w_j d_{Lj} + \sum w_j d_{Uj}$$

for  $j=1$  to  $x$  constraints (Parshall et al., 2001). This is the weighted positive deviation of each constraint from the specified test characteristics. The terms  $d_{Lj}$  and  $d_{Uj}$  are deviations defined relative to all items that could make up the test:

$$q = \sum a_{ij} x_i + a_{tj} + (n - k) v_j$$

Where  $a_{ij} x_i$  is the sum of constraint  $j$  for items currently selected for a test (which can be a 0/1 indicator for content or a psychometric value such as information or proportion correct),  $a_{tj}$  is the value of the constraint for item  $t$  which is not on the current test,  $v_j$  is the average value of constraint  $j$  in the remaining pool, and  $(n-k)$  is the number of unselected items. This sum  $q$  is then compared with the upper and lower bounds of the constraint to compute either  $d_{Lj}$  or  $d_{Uj}$ , with  $d_{Lj}$  computed if the value is less than the lower bound, and  $d_{Uj}$  computed if the value is greater than the upper bound. This equation can be thought of in terms of past (items already selected), present (item under consideration), and future (items in the remaining pool) items.

### Implementing the WDM:

This section describes the algorithm as defined by Swanson and Stocking in 1993; however, actually implementing the algorithm requires some additional considerations. The constraint-specific weights are relevant both to the relative importance of the different constraints, but also to their range. Additionally, program-specific elements such as item use, enemy items, and ranges vs. set values for certain constraints need to be considered.

Using the WDM, test forms are typically built simultaneously, with one item added to each test form over the course of  $L$  iterations (where  $L$  is the test length). Swanson and Stocking (1993) recommended adding a single item at random at the start of the process to ensure that the same pool does not always result in the same forms, although this is not necessary if the sequence of forms varies with each iteration (e.g., add item to form A, then B, then C in first iteration, then C, B, A in second iteration, etc.). An alternative to randomizing the order in which items are added to forms is to order forms based on their relative distance from the targets according to the WDM, or to merely add items to forms in the same order in each iteration. Adding items to forms in the same order in each iteration will result in tests where the first test always receives the best items, and where the same bank/constraints always produce the same test forms.

A constraint relative to content classifications will sum the number of items with that classification. A psychometric constraint will have a much different distribution:  $-0$  to  $1$  for  $p$ -values,  $0$  to  $0.25$  for 1PL information,  $0$  to  $0.5$  for the expected SD of the raw score. Therefore, by default, a content constraint will be four times as important as the most informative item. In order to appropriately apply a weight to reflect the relative importance of the constraint, the values must first be standardized.

The definition of the WDM above will work appropriately if items are drawn without replacement from the item pool. While non-overlapping forms are often desirable, some item banks cannot support that. In situations where some overlap is required across test forms, item usage can be set manually or dynamically to allow for a certain number of test forms. A dynamic process for setting item usage would assign item usage as a function of the minimum number of items needed divided by the number of items in the bank (summed across content constraints). For example, if a minimum of 30 items in content area A are required across two test forms, and the bank contains only 20 unique items in that content area, then item usage can either be set to 2 for all items, or it can be set to 2 for half of the items in that content area. The presence of enemy items can make these calculations more difficult, as the selection of an item with enemies may reduce the available items below the specified minimum. Manually setting item usage may also

take into account the relative exposure of items or other considerations. Once item usage is determined, each potential use of the item should be considered in the pool characteristics calculation.

Enemy items do not fit easily into the WDM, but they do need to be considered in many test builds. Without the capacity to exclude enemy items from test builds, even the most elegant algorithm will not gain acceptance with test editors and SMEs. When an item with enemies is added to a test, the enemies of that item should therefore be removed from the item pool *for that test*. With the items removed from the pool, the calculation of future items will accurately reflect the likely contribution of the pool. Similarly, if item use is greater than 1, other instances of the item should be removed from the pool for that test when the item is selected.

For an ATA, of several necessary decisions about content specifications, some will depend on the characteristics of the test or tests constructed. The WDM may produce test forms that fail to meet specifications when the specifications are unrealistic, or when statistical constraints outweigh content constraints. Certain content domains may have significantly different statistical characteristics (e.g., particularly easy or hard content areas). When testing programs require specific content weights, additional logic outside of the WDM can address the issue of content representation. However, if the content specifications are realistic, the WDM does provide a method for selecting items based on multiple content specifications.

The easiest way to reduce content overrepresentation is to remove items from the available pool for a given test if their content area reaches the maximum. So if content area 1 requires between 8 and 12 items, then remaining content area 1 items are removed from the available pool for test form 1 as soon as the 12<sup>th</sup> item is added to the test. Depending on the number of parallel content constraints, this approach could potentially remove all items from the available pool, and it should ideally target the most important content constraints.

The extent to which the content and statistical constraints are realistic, given the item bank, will determine whether it is possible to build the test forms specified. Moreover, very tight or fixed statistical or content constraints in one area will overwhelm the constraints in other areas—leading to a test that meets no specifications. In fact, especially in situations where item banks are relatively small, it is a good idea to dynamically evaluate the test bank relative to the constraints to see if any forms can be built.

#### **Methods:**

In order to investigate the effects of the different options for implementing the WDM, both simulated and real data were used to generate test forms under targeted conditions. Different conditions for operationalizing the WDM as well as item bank characteristics were varied to investigate the effect of these options.

In theory, adding items to tests in a fixed order (e.g., A1, B1, C1; A2, B2, C2; etc.) will result in form A having better characteristics than form C. Provided tests are built simultaneously and there are relatively few test forms, this effect should be minimal. However, if a large number of forms are built, the effect will be more pronounced. A live pool of 570 items was used to produce tests under random, sequential, and most off-target orders. For each condition, 3 150-item exams, 3 50-item exams, and 10 50-item exams were created. The statistical constraint for all tests was peaked information under the Rasch model at 1.5 logits, although information targets were determined dynamically relative to the item bank characteristics.

The utility of the WDM for selecting items based solely on content was evaluated under several circumstances. Three simulated item banks with two content areas, one with 4 classifications and one with 5, were created. In the first bank, each classification was drawn with equal probability (25% for first area, 20% for the second). For the second pool, not all items were classified on content area 2 (classifications from 0 to 5 were drawn with equal probability). Finally, in the third bank content area, 2 was nested in classifications 1 and 2 of content area 1. Test specifications

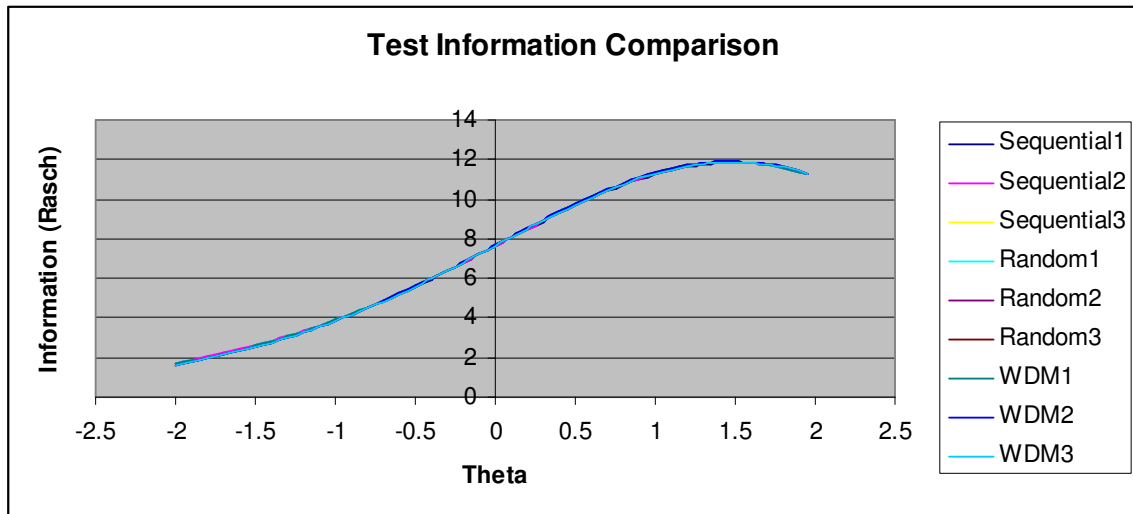
involving both fixed numbers of items and ranges were used to create 10 iterations of 3 test forms.

**Results:**

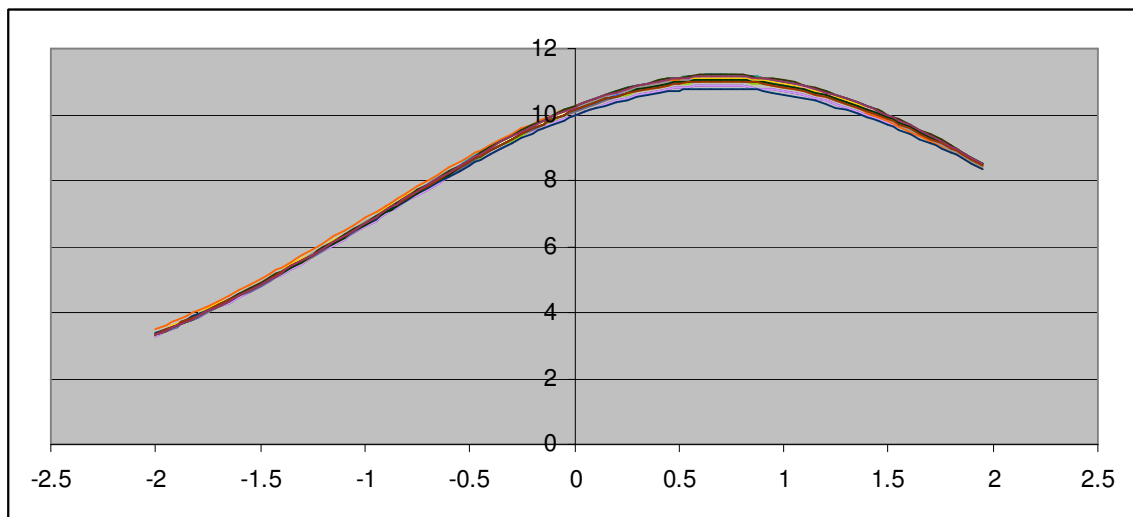
*Order of selection*

Figures 1 and 2 show the results of random, sequential, and most off-target ordering of 3 and 10 test forms built simultaneously. There does not appear to be any difference between the methods, even when relatively many test forms are built sequentially. Further research involving additional constraints is necessary to determine if the order of selection matters when content is considered. Of note for Figure 2 is the shift in peak information away from the target of 1.5 due to the near-full use of the item bank (500 out of 570 items)

**Figure 1. Order of Selection for 3 50-item Tests**



**Figure 2. Order of Selection for 10 50-item Tests**



*WDM and content constraints*

Table 1a shows the number of items in simulated item bank 1 by content classification, while Table 1b shows the characteristics of a test built with fixed content specifications. All tests built with the WDM met the specifications for both content areas.

**Table 1a. Characteristics of Item Bank with Even Distribution of Content**

Count of cc1	cc2					Grand Total
cc1	1	2	3	4	5	
1	10	20	11	16	19	76
2	12	21	7	17	14	71
3	16	19	10	12	16	73
4	15	17	14	17	17	80
Grand Total	53	77	42	62	66	300

**Table 1b. Results of 10 Iterations for Fixed Content Specifications**

content1	spec	min	max	content2	spec	min	max
1	10	10	10	1	6	6	6
2	8	8	8	2	8	8	8
3	10	10	10	3	10	10	10
4	12	12	12	4	6	6	6
				5	10	10	10

In order to test the limits on the WDM, relatively difficult content specifications were used to build test forms from simulated item bank 2. Table 2a shows the characteristics of bank 2, while Table 2b shows the content specifications and minimum/maximum counts of items from each content area over 10 iterations. As with the fixed content specifications, all iterations fall within the specified ranges (this was also the case with the same content specification with bank 1).

**Table 2a. Characteristics of Item Bank with Even Distribution of Content – Missing Content Codes**

Count of cc2	cc2						Grand Total
cc1	0	1	2	3	4	5	
1	10	16	10	7	13	22	78
2	12	9	16	10	11	11	69
3	11	13	15	18	8	12	77
4	11	12	15	16	10	12	76
Grand Total	44	50	56	51	42	57	300

**Table 2b. Results of 10 Iterations for Variable Content Ranges**

content1	spec	min	max	content2	spec	min	max
1	20-25	20	20	1	10-15	10	10
2	0-5	2	4	2	15-20	15	15
3	5-10	7	10	3	0-5	1	4
4	5-10	6	9	4	0-5	4	5
				5	5-10	5	7

Finally, Table 3a shows simulated item bank 3 in which content area 2 is nested within codes 1 and 2 of content area 1. Content specifications were designed such that the minimum number of

items in content area 1 codes 1 and 2 would be insufficient to meet the minimum requirements for content area 2. Table 3b shows the results of 10 iterations of 3 test forms, all of which fall within the specified ranges. It is interesting to note that all of content area 2 item counts are at the minimum.

**Table 3a. Characteristics of Item Bank with Even Distribution of Content – Nested and Missing Content Codes**

Count of cc2	cc2						Grand Total
cc1	0	1	2	3	4	5	
1	5	7	15	9	11	12	59
2	10	14	17	15	10	19	85
3	74						74
4	82						82
Grand Total	171	21	32	24	21	31	300

**Table 3b. Results of 10 Iterations for Variable Content Ranges**

CC1	spec	min	max	CC2	spec	min	max
1	18-20	18	19	1	6	6	6
2	10-15	12	13	2	5-7	5	5
3	0-5	3	5	3	4-8	4	4
4	5-10	5	5	4	7	7	7
				5	8-12	8	8

**Theoretical Significance:**

The evaluation of a simple, easy to implement ATA algorithm in terms of the quality of tests or pools produced offers support for heuristic models. The WDM can theoretically be operationalized in a short period of time for a specific program, and it will provide tests of better statistical quality than can be produced by hand. This research provides information on the efficacy of the WDM in controlled situations; additional research can address the many possible interactions between item bank characteristics and the implementation of a particular ATA algorithm.

**References:**

- Lin, C.-J. (2008). Comparisons between classical test theory and item response theory in automated assembly of parallel test forms. *Journal of Technology, Learning, and Assessment*, 6(8). Retrieved March 24, 2009 from <http://www.jtla.org>.
- Luecht, R. M. (1998). Computer-assisted test assembly using optimization heuristics. *Applied Psychological Measurement*, 22, 224-236.
- Parshall, C. G., Spray, J. A., Kalohn, J. C., & Davey, T. (2001). *Practical considerations in computer-based testing*. New York: Springer-Verlag.
- Swanson, L., & Stocking, M. L. (1993). A method and heuristic for solving very large item selection problems. *Applied Psychological Measurement*, 17, 151-166.
- van der Linden, W. J. (1998). Optimal assembly of psychological and educational tests. *Applied Psychological Measurement*, 22, 195-211.
- van der Linden, W. J. (2005). *Linear models for optimal test design*. New York: Springer.