Open access • Journal Article • DOI:10.1177/014662169201600408

Polynomial algorithms for item matching — Source link

Ronald D. Armstrong, Douglas H. Jones

Published on: 01 Jan 1992 - Applied Psychological Measurement (Sage Publications)

Topics: Polynomial, Matching (statistics), Item analysis and Test theory

Polynomial Algorithms for Item Matching

Ronald D. Armstrong and Douglas H. Jones

Rutgers, The State University of New Jersey

To estimate test reliability and to create parallel tests, test items frequently are matched. Items can be matched by splitting tests into parallel test halves, by creating T splits, or by matching a desired test form. Problems often occur. Algorithms are presented to solve these problems. The algorithms are based on optimization theory in networks (graphs) and have polynomial complexity. Computational results from solving sample problems with several hundred decision variables are reported.
Index terms: branch-and-bound algorithm, classical test theory, complexity, item matching, nondeterministic polynomial complete, parallel tests, polynomial algorithms, test construction.

Gulliksen (1950/1987) presented a matching procedure to estimate split-half test reliability (see Allen & Yen, 1979, pp. 78–83). In the broader context of test construction, splitting a test into two parallel parts can be thought of as the construction of two tests. Van der Linden & Boekkooi-Timminga (1988) proposed several 0–1 mathematical programming models for problems that arise when creating multiple tests. Armstrong, Jones, & Wu (1992) also generated multiple tests based on item matching, with distances based on either classical item statistics or item response theory parameters. They attempted to generate parallel tests with respect to psychometric and content characteristics.

All of the resulting problems presented by Van der Linden & Boekkooi-Timminga (1988) are NP-hard. A problem is NP-hard if there is a nondeterministic polynomial complete (NP-complete) problem that can be polynomially reduced to it. This implies that, in the worst case, the stated problems are difficult to solve. Moderate-size practical problems may take excessive computer time. Van der Linden and Boekkooi-Timminga did not provide computational results, but in two related articles that apply 0–1 programming models (Adema & Van der Linden, 1989; Boekkooi-Timminga, 1990), some computational information was given. For example, to generate a 20-item test from a 500-item databank on a DEC 2060 using the computer program LANDO (Land & Powell, 1973)—which uses a branch-and-bound algorithm procedure—Adema & Van der Linden noted that "it takes too much time to find a 0–1 solution" (p. 284). Boekkooi-Timminga also used LANDO to generate 10-item tests from a 100-item databank. Between 32.7 and 94.7 CPU seconds were required to obtain a 0–1 solution that was not guaranteed to be optimal.

Polynomial algorithms solve problems with solution times that can be expressed as a polynomial in the problem parameters. The notation $O(\cdot)$ is commonly used to denote the order of the polynomial describing the complexity (see Garey & Johnson, 1979, for further information on the concept of complexity). Polynomial algorithms generally are considered "efficient" algorithms. Applying a polynomial algorithm rather than an exponential algorithm often provides several orders of magnitude reduction in computational time. For large problems, in the worst case, polynomial algorithms will outperform exponential algorithms in terms of solution time. In some practical settings, however, exponential algorithms can outperform polynomial algorithms. For example, Bixby, Gregory, Lustig, Marsten, & Shanno (1992) reported excellent results combining interior point and

APPLIED PSYCHOLOGICAL MEASUREMENT
Vol. 16, No. 4, December 1992, pp. 365-371

0146-6216/92/040365-07$1.60

simplex methods to solve practical large-scale linear programming problems. This paper presents polynomial algorithms that are used to solve selected problems in test theory and provides computational results that demonstrate the benefits of these algorithms.

Pairwise Item Matching

Distance Measures

Let n be the number of items in an item bank and d_{ij} be a non-negative measure of the distance between every (i,j) pair of items. δ_{ij} as used by Van der Linden & Boekkooi-Timminga (1988) is an example of a distance measure based on classical item statistics. Other examples of distances, which are derived from item response theory statistics as well as classical item statistics, are given in Armstrong et al. (1992). Two other distance measures based on the three-parameter logistic function are described here.

Let θ_k be the scalar of the trait level for the kth examinee. a_i, b_i, and c_i are the scalars of the true discrimination, difficulty, and the guessing parameters, respectively, for the ith item. Let u_{ik} be the response of the kth examinee on the ith item, which is either 0 or 1. Then $P_{ik} = P(u_{ik} = 1 \mid \theta_k)$, which is the item response function of item i given θ_k and is defined as

$$P_{ik} = P_i(\theta_k) = c_i + (1-c_i)\frac{\exp[Da_i(\theta_k - b_i)]}{1 + \exp[Da_i(\theta_k - b_i)]}, \quad (1)$$

where D is a scaling factor that equals 1.7. The item information function (IIF) is associated with the item response function and is defined as

$$I_i(\theta_k) = \frac{(\partial P_{ik}/\partial\theta)^2}{P_{ik}(1 - P_{ik})}. \quad (2)$$

See Lord (1980) for further explanation.

The Euclidean distance defines the two distances between item i and item j:

$$d_{ij} = [w_1(a_i - a_j)^2 + w_2(b_i - b_j)^2 + w_3(c_i - c_j)^2]^{1/2}, \quad (3)$$

where $w_j \geq 0$, for $j = 1, 2, 3$ and $w_1 + w_2 + w_3 = 1$. The L_2 distance is defined as

$$d_{ij} = \left[\int [I_i(\theta) - I_j(\theta)]^2 f(\theta) d\theta\right]^{1/2}, \quad (4)$$

where $I_i(\theta)$ is the IIF of item i in the item bank given θ, and $I_j(\theta)$ is the IIF of item j given θ. The $f(\theta)$ is the distribution of θ in the population. The integration is over the θ range. Armstrong et al., (1992) used these distances to generate parallel tests for the Armed Services Vocational Aptitude Battery (ASVAB; Shore, 1989).

Matching Algorithm

To simplify the discussion, n is assumed to be an even value (although n also could be an odd value). The pairwise matching problem is to divide all the items into $n/2$ groups of 2 in which the sum of the distances between the items in the paired groupings is a minimum. This problem was discussed by Van der Linden & Boekkooi-Timminga (1988). They described a 0–1 programming model and proposed that the problem be solved using branch-and-bound methods of mathematical programming. An alternative formulation provides a solution method that has complexity $O(n^3)$, rather than the exponential complexity of a branch-and-bound procedure. Papadimitriou & Steiglitz (1982, pp. 255–266) presented the fundamental concepts related to efficient solution methods for the problem.

Define binary variables x_{ij} to indicate whether or not item i is matched with item j (as Van der Linden & Boekkooi-Timminga, 1988, did in their 0–1 programming approach to Gulliksen's matched item-pair method; see Allen & Yen, 1979, pp. 78–83):

$$x_{ij} = \begin{cases} 1 \text{ if } i \text{ and } j \text{ are paired} \\ 0 \text{ otherwise.} \end{cases} \quad (5)$$

Graphical Model

The problem can be described in terms of a graph. The nodes of the graph correspond to the items, and an arc between two nodes indicates a potential pairing of the associated items. The distance on an arc can be defined as previously discussed. Because the arcs of the graph are undirected, x_{ij} is equivalent to x_{ji}.

Let S_k, $k = 1, 2, \ldots, K$ be the subsets of $\{1, 2, \ldots, n\}$ that have an odd cardinality greater than 1; that is, the subsets represent all odd combinations of three or more items. The cardinality of S_k is denoted by $2s_k + 1$. s_k gives the largest number of pairwise item matchings within the subset. Graphically, S_k can be depicted by $2s_k + 1$ nodes.

The mathematical programming problem in which items are optimally paired can be stated as follows:
Minimize

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} d_{ij} x_{ij} , \tag{6}$$

subject to

$$\sum_{j:j<i} x_{ji} + \sum_{j:j>i} x_{ij} = 1; \; i = 1,2, \ldots , n, \tag{7}$$

and

$$\sum_{(i,j)\in S_k} x_{ij} \leq s_k; \; k = 1,2, \ldots , K, \tag{8}$$

where $x_{ij} = 0$ or 1;
$i = 1, 2, \ldots, N - 1$; and
$j = i + 1, \ldots, n$.

The objective function in Expression 6 minimizes the sum of the distances between the assigned pairs. Equation 7 guarantees that each item is assigned to exactly one pair. Inequality 8 assures that no more than the allowable number of items are assigned from each subset with odd cardinality. The value limitations on the decision variables (presented after Inequality 8) give the binary restrictions.

The reason for including the constraints in Inequality 8 is not obvious; in fact, it may seem harmful to include the constraints because there are $2^{n-1} - n$ constraints of this type. The constraints do have a definite purpose, however. They cause the linear problem with x_{ij} restricted to be in the continuous interval [0, 1] to be equivalent to the problem with the binary restrictions; that is, an algorithm exists that will always provide integer solutions for Equations 6 through 8. That is not the case with the formulation proposed by Van der Linden & Boekkooi-Timminga (1988). The formulation of Gulliksen's (1950/1987) pairwise matching method presented here places it in a well-studied class of problems known generally as nonbipartite weighted matching problems (Papadimitriou & Steiglitz, 1982). An algorithm exists to solve such problems in $O(n^3)$ complexity. The proof of existence of the algorithm is nontrivial (see Papadimitriou & Steiglitz, pp. 255–266) and will not be discussed here. Solution methods are provided by Derigs & Metz (1991).

Efficiency of the Algorithm Versus Branch-and-Bound

A study was performed to compare the practical computational efficiency of a branch-and-bound solution approach with an $O(n^3)$ algorithm. LINDO (Schrage, 1986) was used to implement the branch-and-bound method, and an implementation of a special purpose matching algorithm was obtained from Derigs (1988). The item bank consisted of 510 binary-scored arithmetic reasoning items from the ASVAB (Shore, 1989). Items to be matched were selected randomly from the item bank for $n = 10, 20, \ldots, 100$. Distances were computed as the unweighted Euclidean distance between the classical difficulty and point-biserial item parameters. 10 problems were solved for each value of n using a VAX 8550 with VMS 5.3.

The results are summarized in Table 1. A 60-minute time bound was placed on LINDO, and the optimal solution was not obtained (or verified) by LINDO in some cases. In situations in which LINDO used the 60-minute time limit, a time of 60 minutes was used in computing the reported mean. The problems with $n > 60$ were not solved with LINDO because of computer budget restrictions. Derigs' algorithm significantly outperformed LINDO except for the smaller values of n. The variance of CPU time for LINDO was also large because it uses a search technique that may obtain the optimal solution early or may take unproductive paths. The complete ASVAB databank was matched in slightly over 2 minutes with Derigs' code.

Table 1
Mean and Standard Deviation (SD) of CPU Solution Times (In Seconds) for 10 Trials With n Items for an $O(n^3)$ Polynomial Algorithm and LINDO (Numbers in Parentheses Indicate the Number of Trials in Which the Optimal Solution Was Not Guaranteed With LINDO)

	Polynomial		LINDO Branch-and-Bound	
n	Mean	SD	Mean	SD
10	.73	.04	1.14	.04
20	.84	.04	2.81	2.01
30	1.01	.08	4.67	3.48
40	1.17	.05	228.87	492.82
50	1.39	.05	593.50	792.10
60	1.80	.07	1,774.00 (3)	1,452.00
70	2.18	.09	1,397.00 (3)	1,483.70
80	2.65	.10	3,600.00 (10)	0
90	3.14	.19	[a]	[a]
100	3.77	.25	[a]	[a]

[a]Solutions were not attempted with LINDO.

Multiple Item Matching

The problem of matching multiple items is generally more difficult than the pairwise matching problem. Papadimitriou & Steiglitz (1982) showed that the general multiple matching problem is NP-hard. A variation of the problem is, however, solvable with an algorithm of polynomial complexity.

Semiassignment Model

Consider a multiple item matching problem encountered by a testing organization when one of the following situations arises: (1) a reference test exists with n items or (2) a representative subset of n items is selected from an item bank to represent a "typical" test. Each of these situations provides a target with n points to match when creating new tests.

The solution proceeds as follows. From the bank of N items, create T tests to match the target by assigning T items to each target point. No item in the pool can be used more than once; thus, N must be $\geq nT$. Let d_{ij} be the distance of item i in the pool from the jth target point. The procedure for calculating the distance depends on the problem. If the problem is a special problem type, called the "semiassignment," the distance is calculated by minimizing

$$\sum_{i=1}^{N} \sum_{j=1}^{n} d_{ij} x_{ij} \tag{9}$$

subject to

$$\sum_{j=0}^{n} x_{ij} = 1;\ i = 1, 2, \ldots, N, \tag{10}$$

$$\sum_{i=1}^{N} x_{ij} = T;\ j = 1, 2, \ldots, n, \tag{11}$$

and

$$\sum_{i=1}^{N} x_{i0} = N - nT, \tag{12}$$

where $x_{ij} = 0$ or 1;
$i = 1, 2, \ldots, N$; and
$j = 0, 1, 2, \ldots, n$.

$x_{ij} = 1$ if item i is assigned to target point j, and $x_{i0} = 1$ if item i is not assigned to any target point.

Expression 9 minimizes the total distance of all items from the assigned target point. Equation 10 guarantees that each item is assigned at most once to a target point. The variable x_{i0} takes on a value of 1 when item i is not matched to any target point. Equation 11 forces every target point to have T items assigned to it. Equation 12 balances the system by accounting for all items not assigned to a target point (Equation 12 is redundant and could be omitted).

All algorithms designed to solve Equations 9 through 12 obtain integer solutions through the solution of the continuous problem. An algorithm of complexity $O[N(nT)^2]$ was given by Kennington & Wang (1992). They also provided powerful preprocessing procedures that do not change the complexity of the overall solution method but do greatly reduce the solution time for most practical problems.

A special case of the semiassignment problem arises when $N = n$ and $T = 1$. In this case, every item in the pool is used exactly once and the problem is the classical assignment problem, also called the bipartite weighted matching problem. Several algorithms for the assignment problem (Bazaraa, Jarvis, & Sherali, 1990, chap. 10) have complex-

ity $O(n^3)$.

Theunissen (1986, p. 388) developed a semi-assignment model to create T tests that achieves maximal test information at T θ levels. In this model, the binary variables indicate whether an item is assigned to the test for a specified θ level. Each test must have a specified number of items, and no item can be used more than once.

Multiple Splits

Once the items to be used on the T tests have been optimally selected by solving Equations 9 through 12, the items themselves must still be assigned to the individual tests in a structured manner. Van der Linden & Boekkooi-Timminga (1988) proposed minimizing the maximum variance of each test, which, in the split-half case, makes the variances of each test half as close to each other as possible. Some type of balancing procedure for the T subtests does appear to be the only practical approach to the problem. A model slightly different than the one proposed by Van der Linden and Boekkooi-Timminga is proposed here.

Define I_j, where $j = 1, 2, \ldots, n$, as the index set of the T items assigned to target point j. Let q_i be an item statistic for each item. Typically, the q_is are specified so that their sum for each test measures either the variance of the test score (as in the case of Van der Linden & Boekkooi-Timminga, 1988) or the distance the test is from a target. A binary decision variable x_{it} is defined to be 1 if item i is assigned to test t, and is 0 otherwise. The objective is to minimize the range of the total q_is per test over the T tests. That is, minimize Y subject to

$$\sum_{i \in I_j} x_{it} = 1;\ t = 1, \ldots, T;\ j = 1, \ldots, n, \tag{13}$$

$$\sum_{t=1}^{T} x_{it} = 1;\ i \in I_j;\ j = 1, \ldots, n, \tag{14}$$

$$Z - Y \leq \sum_{j=1}^{n} \sum_{i \in I_j} q_i x_{it} \leq Z + Y;\ t = 1, \ldots, T, \tag{15}$$

$$x_{it} = 0 \text{ or } 1,\ i \in I_j;\ j = 1, \ldots, n;\ t = 1, \ldots, T, \tag{16}$$

and

$$Z \geq 0 \text{ and } Y \geq 0\ . \tag{17}$$

The final value of Y will be 1/2 the range of the total q_i per test, and Z will be the midpoint of the range. Alternatively, the sum of the total q_is per test could be selected by providing a specific value for Z, rather than defining Z as a variable. Equation 13 requires that each test be assigned one item from I_j, and Equation 14 requires that each item be assigned to a test. Expression 15 places an interval about the total q_i per test; because the objective is to minimize Y, the value of Z will be the center of the range and Y will be the deviation of Z from the smallest and largest total q_i per test.

Computational Experience

The problem of balancing the assignment of items to individual tests is a more difficult theoretical problem than the optimal assignment of items to the target points. Van der Linden & Boekkooi-Timminga (1988) suggested a branch-and-bound procedure to solve this problem. A simple polynomial heuristic can be devised, however. Sort the items within set I_j based on the value of q_i. Next, sequentially assign items to the T tests by allocating the item with the largest q_i to the subtest with the smallest current value for the partial total q_i per test, and allocate the item with the next largest q_i to the subtest with the next smallest partial total q_i per test. Continue allocating in this way until the smallest q_i with $i \in I_j$ is allocated to the test with the largest partial total q_i per test. This procedure has $nT \times \log T$ complexity, and the difference between the solution obtained with this heuristic and the optimal solution will be no more than the largest q_i value. The heuristic can be improved, with small computational effort, by an interchange procedure after the initial allocation. An explicit statement of the heuristic is given in Armstrong et al., (1992, appendix). This heuristic, even with constraints that force a representation of a specified number of items from taxonomic groups, comes within less than 1% of the optimal allocation. Computational results with the

multiple item matching and test balancing heuristic (Armstrong et al., 1992) indicated that six tests of 30 items each could be generated from the ASVAB in under 2 minutes using an Intel 80286 CPU and no math coprocessor.

The most apparent technique available to optimally solve Equations 13 through 17 is to use Lagrangian relaxation (Nemhauser & Wolsey, 1988). Expression 15 would be brought to the objective function with Lagrange multipliers, leaving an assignment problem solvable in $O[(nT)^3]$ complexity. Branch-and-bound still would be required, in general, to verify the optimal solution.

Discussion

There is considerable literature on the topic of test construction using mathematical programming techniques. Most of the models address problems belonging to the class of NP-hard problems; thus, there are no polynomial algorithms to solve them. This paper presented polynomial algorithms to solve important problems in test construction. The problems formulated in this paper can be solved on a personal computer even with several hundred test items. Additional constraints on the problems presented here may cause the problems to be NP-hard. The NP-hard problems must be solved with branch-and-bound methods, as suggested by Van der Linden and Boekkooi-Timminga (1988), or some other enumerative technique.

References

Adema, J. J., & Van der Linden, W. J. (1989). Algorithms for computerized test construction using classical item parameters. *Journal of Educational Statistics, 14,* 279–290.

Allen, M. J., & Yen, W. M. (1979). *Introduction to measurement theory.* Monterey CA: Brooks/Cole.

Armstrong, R., Jones, D., & Wu, I. (1992). An automated test development of parallel tests from a seed test. *Psychometrika, 57,* 271–288.

Bazaraa, M. S., Jarvis, J. J., & Sherali, H. D. (1990). *Linear programming and network flows.* New York: Wiley.

Bixby, R. E., Gregory, J. W., Lustig, I. J., Marsten, R. E., & Shanno, D. F. (1992). Very large-scale programming: A case study in combining interior point and simplex methods. *Operations Research, 40,* 885–897.

Boekkooi-Timminga, E. (1990). The construction of parallel tests from IRT-based item banks. *Journal of Educational Statistics, 15,* 129–145.

Derigs, U. (1988) Solving non-bipartite matching problems via shortest path techniques. *Annals of Operations Research, 13,* 225–261.

Derigs, U., & Metz, A. (1991). Solving (large scale) matching problems combinatorially. *Mathematical Programming, 50,* 113–121.

Garey, M., & Johnson, D. (1979). *Computers and intractability: A guide to the theory of NP-completeness.* New York: W. F. Freeman.

Gulliksen, H. (1950). *Theory of mental tests.* New York: Wiley. [Reprinted 1987, Hillsdale NJ: Erlbaum.]

Kennington, J., & Wang, Z. (1992). A shortest augmenting path algorithm for the semi-assignment problem. *Operations Research, 40,* 178–187.

Land, A. H., & Powell, S. (1973). *Fortran code for mathematical programming: Linear, quadratic and discrete.* London: Wiley.

Lord, F. M. (1980). *Applications of item response theory to practical testing problems.* Hillsdale NJ: Erlbaum.

Nemhauser, G., & Wolsey, L. (1988). *Integer and combinatorial optimization.* New York: Wiley.

Papadimitriou, C., & Steiglitz, K. (1982). *Combinatorial optimization: Algorithms and complexity.* Englewood Cliffs NJ: Prentice Hall.

Schrage, L. (1986). *Linear, integer and quadratic programming with LINDO.* Redwood City CA: The Scientific Press.

Shore, W. (1989). *Armed Services Vocational Aptitude Battery: Experimental item pool parameters.* Unpublished raw data.

Theunissen, T. J. J. M. (1986). Some applications of optimization algorithms in test design and adaptive testing. *Applied Psychological Measurement, 10,* 381–389.

Van der Linden, W., & Boekkooi-Timminga, E. (1988). A zero-one programming approach to Gulliksen's matched random subtests method. *Applied Psychological Measurement, 12,* 201–209.

Acknowledgments

Partial support for Douglas H. Jones was provided by the Navy Manpower, Personnel, and Training R & D Program of the Office of the Chief of Naval Research under Contract N00014-87-C-0696. Ulrich Derigs supplied a machine-readable version of the FORTRAN code found in Derigs (1988). Yuhong Fu was responsible for performing the computational testing.

Author's Address

Send requests for reprints or further information to Ronald D. Armstrong, Rutgers Graduate School of Management, Newark NJ 07102-1895, U.S.A. E-mail: armstrong@draco.rutgers.edu.