Should We Stop Developing Heuristics and Only Rely on Mixed Integer Programming Solvers in Automated Test Assembly? A Rejoinder to van der Linden and Li (2016)

Applied Psychological Measurement

2017, Vol. 41(3) 227–240

© The Author(s) 2017

Reprints and permissions:

sagepub.com/journalsPermissions.nav

DOI: 10.1177/0146621617695523

journals.sagepub.com/home/apm

Pei-Hua Chen¹

Abstract

This rejoinder responds to the commentary by van der Linden and Li entitled “Comment on Three-Element Item Selection Procedures for Multiple Forms Assembly: An Item Matching Approach” on the article “Three-Element Item Selection Procedures for Multiple Forms Assembly: An Item Matching Approach” by Chen. Van der Linden and Li made a strong statement calling for the cessation of test assembly heuristics development, and instead encouraged embracing mixed integer programming (MIP). This article points out the nondeterministic polynomial (NP)–hard nature of MIP problems and how solutions found using heuristics could be useful in an MIP context. Although van der Linden and Li provided several practical examples of test assembly supporting their view, the examples ignore the cases in which a slight change of constraints or item pool data might mean it would not be possible to obtain solutions as quickly as before. The article illustrates the use of heuristic solutions to improve both the performance of MIP solvers and the quality of solutions. Additional responses to the commentary by van der Linden and Li are included.

Keywords

automated test assembly, item matching, mixed integer programming, MIP solvers, multiple parallel forms

Using mixed integer programming (MIP) with commercially available solvers such as CPLEX (IBM, 2009) when formulating a problem provides great flexibility, and its general usefulness in optimization problems has encouraged its wide application in diverse fields such as transportation and manufacturing. This is especially true because such commercial programs began to be coupled with the advancement of computer technology in recent years. Although heuristics are usually constructed for specialized problems and require customized programming, they can quickly find a reasonable solution. Writing your own program with any programming language

¹National Chiao Tung University, Hsinchu, Taiwan

Corresponding Author:

Pei-Hua Chen, Department of Management Science, National Chiao Tung University, 1001 University Rd., Hsinchu 300, Taiwan.

Email: paulachen@mail.nctu.edu.tw

you like or choosing commercially available MIP solvers that require some training in optimization and integer programming presents various advantages and disadvantages. It may even be tempting to stop developing heuristics and only rely on MIP solvers to solve test assembly problems. However, before one makes such a strong statement as van der Linden and Li (2016) did, let us look into how the MIP approach works and how solutions found in heuristics can be used to improve the performance of MIP solvers.

MIP Approach Versus Heuristics

In the world of optimization, it is common practice to relate computation time to problem size, which is usually described in terms of the number of constraints and variables (Wolsey & Nemhauser, 1999). According to Stocking and Swanson (1993), item pool sizes range from 300 to 5,000 whereas the number of user-specified constraints ranges from 50 to 300 in test assembly practice. Phone consultations with current practitioners show that the operational item pool size can reach about 15,000 items. Notice that the number of user-specified constraints and item pool sizes are among the factors that determine the number of constraints and variables in MIP. Table 1 shows how the number of constraints and variables varies under different item pool sizes, test length, number of forms, and problem types. Integer programming belongs to the class of NP-complete (nondeterministic polynomial-complete, which is a subset of the NP-hard class), which means that no polynomial time algorithms can be found unless $P = NP$. In other words, $P = NP$ is deemed unlikely, and solution time may grow exponentially as the problem size grows, in the worst case scenario. The statement above that Chen (2016) mentioned does not try to “predict” computation time for an MIP problem, but rather to emphasize its computation intensiveness, which may require faster CPU speed and larger RAM as compared with heuristics.

In fact, an MIP solver is a potentially exhaustive search; in many cases, its search space may be reduced by implementing special techniques. There is no guarantee that these special techniques will reduce the search space significantly enough to make the problem solvable within a reasonable time frame. For example, for a binary integer programming problem such as a test assembly with item pool size N , one may need to enumerate 2^N vectors to find a solution, if none of the special techniques implemented within MIP solvers is effective in reducing the enumerations. In addition to problem size, the number of solutions a solver has to enumerate also heavily depends on the problem structure (or problem type) and the problem data. Unless $P = NP$, there will always be a possibility that an MIP solver may encounter endless searching. Even if the problem size is small, there are still some integer programming problems that require a protracted time period to solve and that still cannot guarantee the obtainment of an optimal or near optimal solution (usually called unsolvable problems). For example, in Chen’s (2016) Study 1, although the test information function (TIF) MIP method had only 795 constraints and 2,502 variables, the solver just could not prove optimality in a reasonable period of time.

The Mixed Integer Problem Library (MIPLIB; Konrad-Zuse-Zentrum für Informationstechnik Berlin, 2016) is an official public website that maintains both MIP problems that are currently solvable with today’s codes and unsolved MIP problems that are still open. Many of the open problems have quite small problem sizes. In other words, there are still many MIP problems that can require hours or even days to obtain an optimal or near optimal solution even with state-of-the-art optimization solvers and high-speed machines (Klotz & Newman, 2013). That is part of the reason why research in the optimization field continues to progress and why optimization software developers continue to update solvers’ performances every year.

To make it more specific, the fact that a test assembly problem is NP-hard means there is no guarantee that one may not run into the worst case scenario of enumerating all the 2^N vectors.

Table 1. Summary Table for Problem sizes, Relative Gap Settings, Solutions Type, and Objective Function Obtained in Three Different MIP Methods Under Different Conditions.

MIP method	Pool size	Test length	No. of forms	No. of constraints	No. of variables	Relative gap settings	Objective function	Solutions type
IM MIP	500	30	5	20,530	75,002	0.0001	4.7393	Optimal
IM MIP	500	30	10	26,060	150,002	0.0100	5.9963	Optimal
IM MIP	1,000	30	5	40,530	150,002	0.0001	3.9315	Optimal
IM MIP	1,000	30	10	51,060	300,002	0.0001	4.7120	Optimal
IM MIP	3,000	30	5	120,530	450,002	0.0001	2.9942	Optimal
IM MIP	3,000	30	10	151,060	900,002	0.0001	3.6093	Feasible
IM MIP	6,000	30	5	240,530	900,002	0.0200	2.4928	Optimal
IM MIP	6,000	30	10	301,060	1,800,002	0.0001	2.9942	Feasible
IM MIP	6,000	60	5	420,680	1,800,002	0.0200	5.8553	Optimal
IM MIP	6,000	60	10	481,360	3,600,002	0.0001	7.1258	Optimal
5-point TIF + TCC	500	30	5	1,055	2,503	0.0200	0.0123	1 hr limit exceed
5-point TIF + TCC	500	30	10	1,610	5,003	0.0200	0.0209	1 hr limit exceed
5-point TIF + TCC	1,000	30	5	1,555	5,003	0.0200	0.0140	1 hr limit exceed
5-point TIF + TCC	1,000	30	10	2,110	10,003	0.0200	0.0192	1 hr limit exceed
5-point TIF + TCC	3,000	30	5	3,555	15,003	0.0200	0.0095	1 hr limit exceed
5-point TIF + TCC	3,000	30	10	4,110	30,003	0.0200	0.0207	1 hr limit exceed
5-point TIF + TCC	6,000	30	5	6,555	30,003	0.0200	0.0138	1 hr limit exceed
5-point TIF + TCC	6,000	30	10	7,110	60,003	0.0200	0.0282	1 hr limit exceed
5-point TIF + TCC	6,000	60	5	6,555	30,003	0.0200	0.0134	3 hr limit exceed
5-point TIF + TCC	6,000	60	10	7,110	60,003	0.0200	0.0386	3 hr limit exceed
13-point TIF + TCC	500	30	5	1,135	2,503	0.0200	0.0377	1 hr limit exceed
13-point TIF + TCC	500	30	10	1,770	5,003	0.0200	0.0869	1 hr limit exceed
13-point TIF + TCC	1,000	30	5	1,635	5,003	0.0200	0.0449	1 hr limit exceed
13-point TIF + TCC	1,000	30	10	2,270	10,003	0.0200	0.0978	1 hr limit exceed
13-point TIF + TCC	3,000	30	5	3,535	15,003	0.0200	0.0705	1 hr limit exceed
13-point TIF + TCC	3,000	30	10	4,270	30,003	0.0200	0.1470	1 hr limit exceed
13-point TIF + TCC	6,000	30	5	6,635	30,003	0.0200	0.1058	1 hr limit exceed
13-point TIF + TCC	6,000	30	10	7,270	60,003	0.0200	0.1838	1 hr limit exceed
13-point TIF + TCC	6,000	60	5	6,635	30,003	0.0200	0.2621	3 hr limit exceed
13-point TIF + TCC	6,000	60	10	7,270	60,003	0.0200	0.6175	3 hr limit exceed

Note. MIP = mixed integer linear programming; IM = item matching; TIF = test information function; TCC = test characteristic curve.

The examples that van der Linden and Li (2016) chose may not have run into that problem, but that does not mean a different test assembly problem or set of item pool data will not.

Here is one reason why heuristics are useful in practice when one deals with NP-hard problems. If one relies solely on MIP, one might make a small change to the model or only to the data over which one optimizes, and suddenly, the solution time can explode. This is because the solver has to enumerate those 2^N solutions if none of the special techniques works, and one may just get “unlucky” because of these changes—it may no longer be possible to find a feasible or optimal solution as quickly as before. Table 3 shows how a slight change to the model or data (e.g., adding a cognitive constraint, number of forms, test length, or different item pool sizes) can dramatically alter the running time of the MIP method. In the condition of assembling five forms in item pool sizes of 500 and 1,000, adding a cognitive constraint does not change the running time much. However, the running time does change from 2,109.1 s to 47,117.8 s after adding the cognitive constraint in the condition of assembling five 60-item forms in a 6,000 item pool. Since the MIP method running time greatly depends on the problem type, constraints, and item pool data structure, it implies that each MIP problem is unique and that the running time thus varies case by case. It is hard to tell how often the running time will become too long after certain changes are made to the model or data. One may just add a constraint or change the value of the constraint or use a new item pool data. It is then likely that such a slight change may make the finding of solutions extremely difficult, and therefore it may even take ages to find a feasible solution.

On the other hand heuristics are usually polynomial time algorithms and require less computation time in general. Furthermore, solutions obtained from heuristics can be used in MIP solvers such as CPLEX to jump-start its search. Even if heuristics fail to satisfy all constraints, partially feasible solutions obtained from heuristics can also be used as a jump-start to speed up MIP solvers. An illustration of how solutions obtained from heuristics can improve the performance of MIP solvers will be described in the next section.

Using Solutions Obtained From Heuristics to Improve MIP Solvers' Performance: An Example

The author ran a test assembly problem using the data from Chen's (2016) Study 3. The item pool consisted of 6,000 items, and five 30-item parallel forms were required. The author used conditions including content constraints, cognitive constraints, response time restraints, and enemy items as used in Study 3. In addition, the constraint was added of a maximum of five overlapping items among different forms with none of the items appearing in more than two forms. CPLEX 12.6 was used in an Intel i7-3700 3.4 GHz CPU on a 16GB RAM desktop computer. With the item matching (IM) objective function and the test assembly requirement described above, the IM MIP had 3,666,555 constraints and 1,050,002 variables. CPLEX took 30 min and 39 s to obtain an optimal solution with the relative gap setting of 2%. The solution gap to the best optimal solution possible was 0.67%. Using the solution obtained from heuristics (Chen & Huang, 2016a) to initiate the jump-start, CPLEX took only 3 min and 36 s to obtain an optimal solution with a zero solution gap to the best optimal solution possible. Both the running time and the quality of the solution were thus significantly improved.

Response to Comments on Study I

Chen's (2016) Study 1 was replicated here. The IM MIP objective function was replaced by the sum of interitem distances between one form and the target test as the new objective function,

to obtain better direct control of the form-to-target distances per form. The results of Chen's Study 1 suggest that the commonly used TIF MIP method only considers TIFs and does not yield similar test characteristic curves (TCCs); therefore, an additional test equating procedure is required. The TIF MIP method was replaced by TIF + TCC MIP to address the practical requirements in TCCs and TIFs. The maximum value of the sum of TIF deviations and TCC deviations from the target form at 13 ability points was used as the objective function for TIF + TCC MIP. The TIF and TCC results for each of the five item selection methods are shown in Figure 1 and Figure 2. TIF + TCC MIP performed better in terms of its TIFs and TCCs because its objective function addressed the two criteria directly. The results of the IM MIP and IM heuristics all look quite similar in terms of hitting the TIFs and TCCs.

In Chen's (2016) Study 1, the purpose of adding the D into Equations 9 to 13 was to reduce form-to-form differences. Since D cannot be larger than 2 times y , D can be dropped from the objective function. Even if D in Equations 9 to 13 was later dropped from the objective function to make it easier to solve, the solver still could not prove to be optimal within 30 min.

The IM Heuristics, IM MIP, and TIF MIP

The purpose of the Chen (2016) study was to propose an IM heuristic and compare it with Armstrong's IM heuristic. Chen also wanted to replicate Armstrong's IM MIP method to see how the running time could be improved after 20 years of modern advances in computer technology. It was not the main purpose of the Chen study to compare the IM heuristic and TIF MIP method. That is why the author only included the TIF MIP in the first study. Adding the results of the TIF MIP method served three purposes. First, because TIF MIP is a common practice in test assembly, it was used as a baseline comparison for the IM approach. Second, given the test assembly problem and data used in Chen's Study 1, the results showed that the TIF MIP could not provide an optimal solution within a realistic time period, and additional parameter setting tuning is required to obtain a feasible solution. Third, comparing the TCC results highlighted the strength of an IM approach which fits the TIF and TCC by mimicking item parameters distributions (Armstrong, Jones, & Wu, 1992). Matching item parameters directly produced strictly parallel test forms (Lord, Birnbaum, & Novick, 1968), whereas forms produced by matching TIF were considered weakly parallel (Samejima, 1977). In item response theory (IRT), TCC relates the IRT ability to a number-correct true score and is used when equating true scores. When automated test assembly can produce parallel forms with similar TCCs and TIFs, it implies that forms are psychometrically parallel and no extra equating procedure is necessary.

The IM MIP method and IM heuristics are considered to be the same kind of problem because the purpose of each is to reduce interitem distances from a seed test. On the other hand, the TIF MIP method uses a totally different objective function, and it is therefore not the same kind of problem as IM. The author does not agree that the IM MIP method is inefficient when compared with the TIF MIP method. These are simply different problems with different objective functions.

Of course, using certain points in TIF and TCC as objective functions or item-to-target distances (IM) shows major differences in problem size in terms of the number of constraints and variables. Table 1 summarizes the problem size, objective functions, and relative gap settings for three different methods: IM MIP, 5-point TIF + TCC MIP, and 13-point TIF + TCC MIP. The huge differences in problem sizes between TIF MIP and IM MIP may make the latter look like an undesirable model. The author ran 13-point and 5-point TIF + TCC MIP methods with the data and test assembly constraints used in Chen's (2016) Study 3. Table 2 summarizes the total running time for the R-R-B, R-MD-B, IM MIP, 5-point TIF + TCC MIP, and 13-point

Figure 1. Test information functions for the five item selection methods.
Note. MIP = mixed integer linear programming.

Figure 2. Test characteristic curves for the five item selection methods.
Note. MIP = mixed integer linear programming.

Table 2. The Total Running Time (in Seconds) of the Five Item Selection Methods Under Different Conditions.

Pool size	500		1,000		3,000		6,000		6,000	
Test length	30		30		30		30		30	
No. of forms	5	10	5	10	5	10	5	10	5	10
R-R-B ^a	4.77	5.66	15.74	19.59	157.81	172.64	628.26	744.34	1,055.00	1,055.08
R-MD-B ^a	4.75	6.85	14.90	21.43	141.63	181.40	626.90	754.09	1,098.87	1,099.68
IM MIP	107	93 ^b	295	8,169	5,636	24,328	4,050	4,659	6,954	19,505
5-point TIF + TCC ^c	3,608	3,682	3,610	3,614	3,609	3,623	3,768	3,667	11,208	10,880
13-point TIF + TCC ^c	3,742	3,606	3,618	3,607	3,614	3,619	3,621	3,665	10,822	10,863

Note. IM = item matching; MIP = mixed integer linear programming; TIF = test information function; TCC = test characteristic curve; R-R-B = Random item sequence assigned to random form with content balancing; R-MD-B = Random item sequence assigned to maximum distance form with content balance.

^aThe running time of the two heuristics methods was recorded after 100 successful trials were obtained.

^bUsed a larger relative gap of 1% to obtain solutions more quickly.

^cThe 5-point and 13-point TIF + TIF MIP's running times were preset to 1 hr to 10,000 s, and all stopped after reset time limit exceeded.

TIF + TCC MIP. The IM MIP used the revised objective function described earlier. It can be seen that for the same kind of test assembly specifications, in general, the IM MIP took more time in solving the problem than did the two heuristics.

However, the modification of objective function for the IM MIP does not change the fact that IM MIP takes more time to find a solution than the R-R-B or R-MD-B because of its computational extensiveness nature. Under almost all conditions for the 5-point TIF + TCC MIP and 13-point TIF + TCC MIP, the solver could not close the gap between the best feasible solution and the best possible optimal solution within 1 or 3 hrs. In short, problems were unsolvable within a reasonable time. Setting up the relative gap to be 2% as van der Linden and Li (2016) suggested did not work at all in this case because the best possible optimal solution is zero. In this case, one instead needs to set up an acceptable absolute gap based on his or her past experience. The current default setting for absolute gap is 10^{-6} , and the default setting for relative gap is 10^{-4} in CPLEX. In contrast, most conditions using the IM MIP method yielded optimal solutions with the same data and test assembly constraints as in Chen's (2016) Study 3. In other words, in the case with a smaller problem size but more difficult problem, TIF + TCC MIP was not solvable, whereas in the case with a larger problem size (more constraints and variables), IM MIP could find an optimal solution within realistic time under most conditions with default settings. This is just another example of how MIP could miserably fail to prove optimal in some small instances.

Response to Comments on Study 2

Due to the page limitation for articles published in *Applied Psychological Measurement*, the table of the total running time of the five item selection methods in Study 2 was deleted in Chen (2016). Those results are shown in Table 3 as requested. Readers should refer to the discussions of the results in Chen's Study 2.

Response to Comments on Study 3

The purpose of Chen's article is not to provide a general purpose heuristic approach that seeks to solve every test assembly problem. Chen's (2016) Study 3 added some of the most

Table 3. The Total Running Time (in Seconds) of the Five Item Selection Methods Under Different Conditions in Chen's (2016) Study 2.

Pool size	500				1,000				3,000				6,000			
	30		60		30		60		30		60		30		60	
Test length	30		60		30		60		30		60		30		60	
Number of forms	5	10	5	10	5	10	5	10	5	10	5	10	5	10	5	10
Adding cognitive	No	Yes	No	Yes	No	Yes	No	Yes	No	Yes	No	Yes	No	Yes	No	Yes
R-R-B	2.4	2.4	2.3	2.4	7.1	7.2	7.2	7.2	52.0	52.1	52.0	52.2	201.3	199.7	200.3	202.0
R-MD-B	2.4	2.4	2.4	2.5	7.2	7.3	7.2	7.3	52.6	52.6	52.6	53.5	202.1	201.4	201.8	201.8
R-MD-R	13.5 ^a	32.2	32.2	31.1 ^a	28.0	28.0	28.0	28.0	322.1 ^a	322.1 ^a	322.1 ^a	350.5 ^a	350.5 ^a	350.5 ^a	651.6 ^a	651.6 ^a
R-R-R	12.2 ^a	30.3	30.3	27.2 ^a	30.6	30.6	30.6	30.6	296.0 ^a	296.0 ^a	296.0 ^a	346.3 ^a	346.3 ^a	346.3 ^a	646.1 ^a	646.1 ^a
IM MIP	29.5	29.4	1,842.9	1,850.0	63.0	66.0	214.0	1,023.7	417.0	504.2	3,540.8	4,578.9	711.0	726.6	1,584.3	3,781.2

Note. The running time of the four heuristics methods was recorded after 100 successful trials were obtained. IM = item matching; MIP = mixed integer linear programming.

R-R-R = Random item sequence assigned to random form with Armstrong's replacement; R-MD-R= Random item sequence assigned to maximum distance form with Armstrong's replacement.

^aIndicates that the algorithm ran for 1,000 infinite loops and could not obtain any solution.

^bIndicates that the CPLEX could not find any feasible solutions and stopped due to running out of memory while solving the model.

commonly used constraints in practice, such as content and cognitive constraints, enemy items, and response time. The item overlap was set to zero because, in most cases, test forms are usually linked by means of prechosen anchor items by psychometricians and content experts. Letting the solver (or robot) decide the anchor items may not be desirable in practice.

Actually, heuristics with a form-to-form overlap with the additional requirement that common items cannot appear in more than two forms have been developed (Chen & Huang, 2016a) on top of Chen (2016). In response to van der Linden and Li (2016), the R-R-B and R-MD-B item selection algorithms in Chen (2016) regarding content, cognitive constraints, enemy items, and response time are described in the appendix. For dealing with passage constraints such as set-based items, Chen and Huang (2016b) extended the IM heuristic method to incorporate complex test specifications, including set-based item constraints. In addition, the IM approach has recently been implemented in a large-scale Chinese proficiency test to assemble 50 forms (Wang, Zheng, Zheng, Su, & Li, 2016).

Response to Other Comments

Is Matching 3 to 5 Ability Points Enough?

Van der Linden and Li (2016) stated that matching 3 to 5 ability points to fit TIFs and TCCs could produce satisfactory results. However, the author used Study 1 item pool and test specifications and matched the target test's TIFs and TCCs on 3 (−1, 0, 1), 5 (−2, −1, 0, 1, 2), 7 (−2.5, −2, −1, 0, 1, 2, 2.5), and 13 (−3, −2.5, −2, −1.5, −1, −0.5, 0, 0.5, 1, 1.5, 2, 2.5, 3) ability points. TIF MIP and TIF + TCC MIP were both conducted. Each of the conditions was set up to run for 30 min. The results shown on Figure 3 and Figure 4 indicate that at least 5 to 7 ability points must be matched to produce smooth TIFs and TCCs across the ability ranges between [−3, +3].

Optimizer Time Limit

Van der Linden and Li (2016) mentioned that the default optimizer time limit in CPLEX is 1 min. According to the IBM knowledge center (IBM, 2016) and the CPLEX software, the optimizer time limit is 10^{75} s in CPLEX Versions 12.4, 12.5, and 12.6.

Conclusion

In summary, neither the heuristics approach nor the MIP approach should be intended to completely replace the other. In fact, heuristics could serve to complement MIP methods when MIP solvers are having difficulty solving a problem within reasonable time. When MIP solvers can quickly find good solutions, one does not need to develop heuristics. However, when MIP problems become unsolvable, it makes sense to develop heuristics to quickly find a feasible solution. In addition, one can use heuristics to obtain a feasible solution while reducing the running time and increasing the quality of solutions found by MIP solvers. Even partially feasible solutions obtained from heuristics can be used in MIP solvers. In other words, solutions obtained from heuristics can be used to improve the performance of MIP solvers.

With advancements in optimization and computational technology, MIP solvers have been widely used in various fields. In an ideal world, users can simply formulate their problem as an optimization problem, write some syntax, and use MIP solvers like a black box to obtain solutions. However, MIP approach may not always provide solutions while users do not have access to the source code, and other approaches should also be considered (Chang, 2007). The author appreciates the opportunity for further discussion. Discussions in Chen (2016), van der Linden

Figure 3. TIFs for the eight MIP conditions.
Note. TIF = test information function; MIP = mixed integer linear programming; TCC = test characteristic curve.

Figure 4. TCCs for the eight MIP conditions.
Note. TCC = test characteristic curve; MIP = mixed integer linear programming; TIF = test information function.

and Li (2016), and this article have clearly shown that users could not use MIP solvers without any basic knowledge and training in integer programming. Practitioners should be trained to know how to adjust various parameter settings when they run into problems such as the ones that are unsolvable within a reasonable time period or that cause the platform to run out of memory. Test assembly problems are considered difficult MIP problems, and readers who want to know more about solving such problems could refer to practical guidelines for solving difficult MIP problems (Klotz & Newman, 2013).

Appendix

R-R-B and R-MD-B Item Selection Algorithm With Content, Cognitive, Enemy Items, and Response Time Constraints

1. Prepare a sorted pool-to-target interitem distance matrix in ascending order.
2. Randomly pick a target item to start the item selection process.
3. The item with the smallest interitem distance in the pool-to-target matrix for the corresponding target item has a higher priority to be considered as a candidate item.
4. Before an item i within content area j is assigned to a form, it is necessary to calculate the accumulated number of items for each form in content area j and compare the values with the lower and upper bounds of content area j . The required lower bounds of content area j in all the F forms have to be fulfilled first without exceeding the upper bounds. The candidate item i within content area j will only be assigned if not all of the F forms have reached the upper bounds of content area j . Otherwise, go to Step 3 and evaluate another candidate item.
5. The same method in Step 4 will be used to deal with cognitive constraints.
6. The response time of the candidate item will be compared with C_{RT} , the average of the upper and lower bounds of response time constraints per item ($C_{RT} = (LB_{RT} + UB_{RT})/2N$). If the item response time value of the candidate item is larger than C_{RT} , then the next selected item's response time should be less than C_{RT} and vice versa. Otherwise, go to Step 3 and evaluate another candidate item.
7. Before an item is assigned to a form, check whether the candidate item is on the enemy item list. If the answer is yes, then the other enemy items will be marked to prevent further selection to the same form.
8. For the R-MD-B method, a higher priority is given to the form f with the larger sum of the total interitem distances between the target form and form f when assigning item i . For the R-R-B method, items will be randomly assigned to one of the F forms as long as adding an item satisfies the content area requirement during the item assignment process.
9. The item selection process described above will be repeated until all T items have been selected for each form.
10. The total response time constraints for each form will be evaluated. If any of the total response times of the forms do not fall within LB_{RT} and UB_{RT} , the set of forms will be discarded, and start over.

Acknowledgments

The author would like to thank two anonymous reviewers and the editor-in-chief for their helpful comments on earlier versions of this article. The author is grateful for the time and care given by those

colleagues during phone consultations. Thanks also goes to Hsu Chia-Wei and Huang Cheng-Yi for their assistance in running programs. Special thanks to Daniel Junglas for his support and assistance.

Declaration of Conflicting Interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

References

- Armstrong, R. D., Jones, D. H., & Wu, I. L. (1992). An automated test development of parallel tests from a seed test. *Psychometrika*, 57, 271-288.
- Chang, H.-H. (2007). [Review of the book *Linear models for optimal test design*, by W. J. van der Linden]. *Psychometrika*, 72, 279-281.
- Chen, P.-H. (2016). Three-element item selection procedures for multiple forms assembly: An item matching approach. *Applied Psychological Measurement*, 40, 114-127. doi:10.1177/0146621615605307
- Chen, P.-H., & Huang, C.-Y. (2016a). *Improving the item-matching method for a practical test assembly problem*. Paper proposal accepted at the National Council on Measurement in Education, April 2017, San Antonio, TX.
- Chen, P.-H., & Huang, C.-Y. (2016b). *An item-matching heuristic method for a complex multiple forms test assembly problem*. Paper presented at the National Council on Measurement in Education, April, 2016. Washington, DC.
- IBM. (2009). IBM ILOG CPLEX (Version 12). Armonk, NY: Author.
- IBM. (2016). *Optimizer time limit*. Retrieved from https://www.ibm.com/support/knowledgecenter/SS9UKU_12.5.0/com.ibm.cplex.zos.help/Parameters/topics/TiLim.html
- Klotz, E., & Newman, A. M. (2013). Practical guidelines for solving difficult mixed integer linear programs. *Surveys in Operations Research and Management Science*, 18, 18-32.
- Konrad-Zuse-Zentrum für Informationstechnik Berlin. (2016). *Mixed integer problem library*. Retrieved from <http://miplib.zib.de>
- Lord, F. M., Birnbaum, A., & Novick, M. R. (1968). *Statistical theories of mental test scores*. Oxford, UK: Addison-Wesley.
- Samejima, F. (1977). Weakly parallel tests in latent trait theory with some criticisms of classical test theory. *Psychometrika*, 42, 193-198.
- Stocking, M. L., & Swanson, L. (1993). A method for severely constrained item selection in adaptive testing. *Applied Psychological Measurement*, 17, 277-292.
- van der Linden, W. J., & Li, J. (2016). Comment on three-element item selection procedures for multiple forms assembly: An item matching approach. *Applied Psychological Measurement*, 40, 641-649. doi:10.1177/0146621616664075
- Wang, S., Zheng, Y., Zheng, C., Su, Y.-H., & Li, P. (2016). An automated test assembly design for a large-scale Chinese proficiency test. *Applied Psychological Measurement*, 40, 233-237. doi:10.1177/0146621616628503
- Wolsey, L. A., & Nemhauser, G. L. (1999). *Integer and combinatorial optimization*. New York, NY: Wiley Interscience.