A Generalized Objective Function for Computer Adaptive Item Selection

Harold Doran
Testsuhiro Yamada
Ted Diaz
Emre Gonulates
Vanessa Culver

Human Resources Research Organization
Working Draft
Not for Citation
Corresponding author: hdoran@humrro.org

January 12, 2024

Abstract

Computer adaptive testing (CAT) is an increasingly common mode of test administration offering improved test security, better measurement precision, and the potential for shorter testing experiences. This paper presents a new CAT algorithm based on a generalized objective function to support multiple types of testing conditions and principled assessment design. The generalized nature of the algorithm permits a wide array of test requirements allowing experts to define what to measure and how to measure it and the algorithm is simply a means to an end to support better construct representation. This work also emphasizes the computational algorithm and its ability to scale to support faster computing and better cost containment in real-world applications than other CAT algorithms. We make a significant effort to consolidate all information needed to build and scale a CAT so that expert psychometricians and software developers can use this document as a self-contained resource and specification document to build and deploy an operational CAT platform.

Keywords: computer adaptive testing; computational psychometrics; item response theory

Introduction

The main purpose of computer adaptive testing (CAT) is to construct a unique test form for an examinee by adapting test items around their unique level of ability on the construct being measured (Reckase, Ju, & Kim, 2019). In theory, the form that yields the maximum precision for the examinee is one where the test information function (TIF) is maximized at the examinee's true level of ability (Babcock & Weiss, 2009) and is best in the sense that it ensures a psychometric alignment between examinee ability and statistical characteristics of the items (Chang, 2015). Because true ability varies by individual, this implies a unique test form for each examinee where test items are carefully curated by a selection algorithm for administration. When item selection is tailored to examinees, an adaptive test can provide more information regarding the examinee's ability with fewer items relative to a fixed form test (W. J. van der Linden, 2005b; Yao, 2019) and by providing a unique form, test security is improved as examinees in similarly proctored settings will have forms with different test items when item exposure is well-controlled (Yasuda, Hull, & Mae, 2022).

A challenge in adaptive testing is that the true ability is latent and the best indicator of the true ability, the observed ability, changes as examinees provide responses to new items as they progress through a test. In general, few items provide very little information regarding the true ability and more items provide more information (W. J. van der Linden & Pashley, 2010). As a consequence, an adaptive test can iteratively improve its choice of items with a selection algorithm as more information is provided about the examinee to form the best possible test form.

Implicit in the description above is the idea that the term "best" is well-understood for item selection (Kingsbury & Zara, 1989) and that some transparent machinery is available to choose those items from a test bank. In fact, "best" can be used differently depending on the test purpose. Therefore, it's key to begin with an operational definition of "best" and then create a mechanism that chooses the best items to uniquely construct a test form for an examinee.

All adaptive algorithms share the common feature that they adapt items on ability, usually on the individual examinee (W. J. van der Linden, 2005a), but optionally the algorithm may maximize information near a critical point, such as a cutscore for classification. In this way, CATs minimize the individual error variance and provide greater measurement precision. However, adaptive algorithms can vary in how they define "best" in the sense of choosing the items for administration (Ho & Dodd, 2012). For instance, if there is no test blueprint, then "best" might be considered exclusively on the psychometric properties of the items vis-à-vis the individual tester. If there is a content blueprint with requirements to choose items from different dimensions of that blueprint, then perhaps "best" can be defined to include content characteristics of the items. If both factors are important, then the "best" item is one where the algorithm simultaneously considers the measurement properties of the items and their non-statistical content characteristics at each step.

Perhaps the idea of "best" shouldn't be strictly defined by the capabilities of the algorithm. Test designers should follow theories of principled assessment design (Mislevy & Riconscente, 2006) and the algorithm itself should be able to administer any type of test form developers can imagine, a concept that has been formerly well-articulated (Stocking & Swanson, 1993). In this case, "best" is situationally defined and the algorithm is nothing more than a means to an end and is free to select items from any construct-relevant criteria the test developers wish to impose. This generalized flexibility is one means by which the validity claims regarding test scores arising from CATs can be enhanced (Luecht, De Champlain, & Nungester, 1997; Wise, 2023).

There are many examples of CATs used across the education, certification, licensure and selection fields that have large computational demands for high volumes of test takers. The Armed Services Vocational Aptitude Battery is one example for military selection (DMDC, 2008), the Smarter Balanced Consortium deploys millions of adaptive tests across states in the U.S. in educational testing (Cohen & Albright, 2014), and the National Council Licensure Examination uses adaptive tests for licensure and certification for medical professionals (Seo, 2017). Some jurisdictions in the U.S. have even codified CATs as a required mode of test administration for educational testing (Florida State Statute §1008.25, 2023; Virginia State Statute §22.1-253.13:3, 2021).

The vast literature on adaptive testing commonly addresses a particular component within the CAT algorithm, such as item exposure (Huebner, Wang, Quinlan, & Seubert, 2015; Leung, Chang, & Hau, 2002; Sympson & Hetter, 1985), ability estimation (Bock & Mislevy, 1982; T. A. Schmitt & Walker, 2010), information functions (Chang & Ying, 1996), or optimization approaches (Choi, Moellering, Li, & van der Linden, 2016). Less common in the literature are thorough, end-to-end, descriptions of the inner workings of a computer adaptive engine providing enough detail such that an independent team can identify all necessary psychometric and computational components in a complete, self-contained location to build a CAT platform for operational use, although van der Linden (2022) provides a complete example of a shadow test approach and Han (2018) provides a review of many common components included within a CAT algorithm.

Most often, descriptions of CAT algorithms provide a procedural schema describing steps for the algorithm but lack a highly detailed computational algorithm and decision details needed to deploy operational infrastructure (Wainer, Dorans, Flaugher, Green, & Mislevy, 2000). For example, some describe adaptive technologies and their heuristics (Straetmans & Eggen, 1998) but do not provide thorough algorithmic descriptions. Others provide computational details but limit the focus to a particular computational component, such as exposure control or optimal methods for computing psychometric information (Ma et al., 2023). One example provides seemingly complete details for an item selection algorithm using Monte Carlo methods (Belov, Armstrong, & Weissman, 2008), though details on its scalability and computational complexity are not evaluated. Hence, we cannot know the degree to which this is scalable for large-scale design, but Monte Carlo methods are in general very computationally time-consuming.

Purpose and Organization

This paper intends to serve as an end-to-end resource bringing transparency to a new generalized item selection algorithm with a specific emphasis on building a computationally scalable implementation. The highly generalized nature of the algorithm permits test requirements related to content balancing, cognitive complexity demands, item enemy targets, types of items administered, psychometric measurement targets, just to name a few. Some specific examples of what this generalized algorithm can achieve include adapting only on ability to minimize individual error, simultaneously adapt on ability and provide content balancing to match requirements defined in a test blueprint, only control for content requirements in which case the test form is linear-on-the-fly (LOFT), limit the algorithm to administer items from a subset of the bank without bifurcating the bank such as when item enemies might exist, administer a collection of items nested within any common grouping structure, flexibly administer operational and experimental field test items spiraled together, and build on-the-fly multistage adaptive forms. Importantly, the underlying al-

gorithm and its code do not need to be altered to achieve any of these variants as customizability is a function of the algorithm itself.

Our specific aim is to achieve four objectives in this manuscript. First, we propose and fully describe a generalizable CAT algorithm. Second, we include complete end-to-end requirements for deploying this algorithm to support operational CAT implementations. The intent is that this document can serve as a specification document with enough detail allowing an expert psychometric and software development team to build and deploy an operational instance of a CAT by following the guidance contained here. Third, we justify the scalability of the algorithm relative to other approaches and propose ways to implement the algorithm to realize minimal latency at the test-taker level and improved cost-containment in real-world applications. Fourth, we demonstrate the behavior of the algorithm with some test trial cases to document its behavior in simulated settings.

The Generalized Item Selection Algorithm

Let the general item selection function for item $j = \{1, \ldots, J\}$ for examinee $i = \{1, \ldots, P\}$ at step $t = \{1, \ldots, T\}$ needed for test feature $g = \{1, \ldots, G\}$ be

$$\lambda_{jit} = \pi_j I_j^{'}(\widehat{\theta}_{it}) + \sum_{g=1}^{G} \upsilon_g w_{jg} m_{igt} + \sum_{g=1}^{G} h_g w_{jg} \alpha_{igt}, \tag{1}$$

where the function is a weighted linear composite of three additive components on the right-hand side including 1) the item information function, 2) the value of the content balancing parameter needed to fulfill other test developer features, and 3) a stochastic content parameter to randomly boost and potentially vary the administrative sequence of test items by dimension. Table (1) provides an overview and brief description of the notation used in Equation (1). However, each term in this objective function is fully described in complete detail in the remaining sections.

In this context, the term λ_{jit} is a scalar-valued statistic describing the potential for an item to best fulfill the requirements necessary to meet test form criteria for a given examinee at each iteration of the test process. The maximum value of this objective function, $\max \lambda_{jit}$, is the item with the most potential at step t to fulfill the objectives of the test selection algorithm.

Throughout this manuscript, we differentiate between two terms, *algorithmic* details and *decision* details. The term "algorithmic details" is used to mean the specific components in Equation (1) and "decision details" are how those ideas are implemented. For example, the item information function is an algorithmic component of the objective function. How item information is computed is a decision detail. We also use the term *requirements* and not *constraints* when referring to our algorithm. This is to avoid confusion with terms commonly used in mixed integer linear programming (MILP). In MILP, constraints are used to define a feasible region and a solution must exist within that feasible region. Our approach defines boundaries for test features, and they resemble constraints, but optimization is performed differently.

Parameter	Description
π_j	User-defined importance weight associated with information for the jth item.
$I_j^{'}(\widehat{\theta}_{it})$	Item information function for item j evaluated at the ability estimate for examinee i at step t rescaled to be on same unit space as m_{igt}.
υ_g	User-defined importance feature weight associated with dimension g of test blueprint.
w_{jg}	Indicator value linking item j to feature g of a test blueprint.
m_{igt}	Value of content priority index for examinee i at step t for feature g.
h_g	Binary indicator to toggle content sequence parameter on ($h_g = 1$) or off ($h_g = 0$).
α_{igt}	Random draw from a uniform distribution to balance content administration.
$\widehat{\theta}_{it}$	Estimated ability for examinee i at step t.

Table 1: Notation for CAT Algorithm in Equation (1)

Computational Implementation

The scalar-based expression in Equation (1) is useful for general consideration of the idea, but not a useful computational representation. Equivalently, let the objective function be

$$\boldsymbol{\lambda}_{it} = f(\boldsymbol{\pi}, \boldsymbol{I}'(\boldsymbol{\theta}_{it}), \boldsymbol{W}, \boldsymbol{m}) = \boldsymbol{\pi} \odot \boldsymbol{I}'(\boldsymbol{\theta}_{it}) + \boldsymbol{W}\boldsymbol{m}, \tag{2}$$

where the space and dimensions are $\boldsymbol{\pi} = \{\pi_1, \ldots, \pi_J\} \in \mathbb{R}^J_{[0,\infty]}$, $\boldsymbol{I}'(\boldsymbol{\theta}_{it}) = \{I_1^{'}(\widehat{\theta}_{it}), \ldots, I_J^{'}(\widehat{\theta}_{it})\} \in \mathbb{R}^J_{[0,1]}$, $\boldsymbol{W} \in \mathbb{R}^{J\times G}_{[0,1]}$ is an indicator matrix where the rows are the test items in the bank and the columns are values linking the jth item with the gth item feature, h_g is a binary indicator, and $\boldsymbol{m} = \{(\upsilon_1 m_{i1t} + h_1\alpha_{1it}), \ldots, (\upsilon_G m_{iGt} + h_G\alpha_{Git})\} \in \mathbb{R}^G_+$. Here the notation $\mathbb{R}_{[a,b]}$ is used to mean a continuous number on the interval $[a, b]$ including endpoints, $\boldsymbol{\pi} \odot \boldsymbol{I}'(\boldsymbol{\theta}_{it})$ is the Hadamard product yielding a vector of length J, and $\boldsymbol{W}\boldsymbol{m}$ is the ordinary matrix product also yielding a vector of length J. Equation (2) is simply a more concise expression of Equation (1) and in the same way yields a vector of values $\boldsymbol{\lambda}_{it} = \{\lambda_{1it}, \lambda_{2it}, \ldots, \lambda_{Jit}\}$.

Note that $\boldsymbol{W}$ has dimensions $J \times G$, meaning a test item can be linked to many different features. In doing so, it is important to normalize the rows in the matrix $\boldsymbol{W}$ to account for the number of features an item is associated with. Without the normalization, items linked to many features would be more attractive to the item selection algorithm given that the resulting value for λ_{jit} is the row sum over all columns in this matrix. The elements of $\boldsymbol{W}$ before normalization assume values of $w_{jg} = 1$ if item j is associated with feature g and 0 otherwise. Then, the original rows of the matrix $\boldsymbol{W}$, denoted as $\boldsymbol{w}_j = \{w_{j1}, \ldots, w_{jG}\}$, are normalized where each element becomes $w^*_{jg} = w_{jg} / \sum_{g=1}^G w_{jg}$ and $\sum_{g=1}^G w^*_{jg} = 1$.

Equation (2) could treat $\pi_j = \pi \; \forall \; j$ as a constant information weight applied in the same way to all items. However, its treatment as a vector offers broader applications where different weights for information could be applied to vary over some types of items. This is useful in situations where the psychometric properties of items could be exploited as weights. For example, it could be useful to make use of the item-specific variances, σ^2_j, and use their reciprocals as weights

such as $\boldsymbol{\pi} = \{\frac{1}{\sqrt{\sigma_1^2}}, \ldots, \frac{1}{\sqrt{\sigma_J^2}}\}$ mimicking weighting strategies used in survey design for complex samples (Wolter, 1985). Though interesting, some caution on this strategy is needed as this would cause the algorithm to prefer items with smaller standard errors and lead to some potential item overexposure. A second potential use of the vector could be to achieve a quasi a-stratification approach (Huebner, Wang, Daly, & Pinkelman, 2018) where $\pi_j = a_j^{-1}$ is set to the reciprocal of the a-parameter causing the algorithm to use items with smaller slopes more often in early item administration. Last, a third option could be to set $\pi_j = 1$ for operational items and then set $\pi_j = 0$ for experimental field test items. In this way, the algorithm offers an embedded field test mechanism where trial items are randomly seeded alongside operational items.

Optimization Approach

The goal of the objective function is to yield a value denoting which item is best for test administration. Our objective function and test requirements could be formulated as a MILP problem as commonly used in the automated test assembly literature (W. J. van der Linden, 2005c). MILP has been demonstrated for the shadow test method (Choi et al., 2016) and the weighted deviation method (Stocking & Swanson, 1993). However, MILP is computationally expensive, challenging to implement in applied settings, and is not guaranteed to find an optimal solution. Additionally, the mathematical complexity of linear programming almost always requires an external commercial or open-source optimizer (W. van der Linden, 2022) instead of code written and owned by the development team building the CAT engine. This layers in one additional factor development teams need to manage as application programming interfaces (APIs) may change in which case live testing could be interrupted if using a web-based API or code would need to be modified to accommodate API changes.

Here we propose optimization in a different way using a sort-and-search algorithm. In our approach, the input values in $f(\boldsymbol{\pi}, \boldsymbol{I}'(\boldsymbol{\theta}_{it}), \boldsymbol{W}, \boldsymbol{m})$ are treated as known, fixed quantities for computing the elements of $\boldsymbol{\lambda}_{it}$, thus allowing us to directly compute Equation (2) and find the maximum value in the set $\boldsymbol{\lambda}_{it}$, or $\max \boldsymbol{\lambda}_{it}$. When $\boldsymbol{\lambda}_{it}$ is sorted in descending order, the first item in the vector is best for selection in the sense that it contributes best to the objectives defined by the test developer. We later justify the use of a quicksort algorithm via its time complexity and suggest the search space for computing the values in $\boldsymbol{\lambda}_{it}$ can be reduced to achieve improved scalability.

User-Defined Configuration Weights

This section offers a formal definition of the user-defined importance weights. These weights, combined with different ways to configure the content priority index, are what make Equation (2) highly configurable. Pragmatically, they are simply ways to balance the importance between the statistical properties of the items and their non-statistical content characteristics. Examples of their application are provided in the simulation section showcasing how different configurations yield different adaptive test outcomes.

Formally, the values for π_j and υ_g are scalar-valued importance weights defined as $\mathbb{R}_{0 \leq \omega \leq \infty} = \{\omega \in \mathbb{R} | 0 \leq \omega \leq \infty\}$ which is used to mean importance weights in the set $\omega \in \{\pi_j, \upsilon_g\}$ can be any positive real number, including 0. Note that importance weights of $\pi_j = 0 \; \forall \; j$ cause for the algorithm to be either driven entirely by content requirements (linear-on-the-fly) and the most

primitive form of an adaptive algorithm is a special case of Equation (1) when $\{\upsilon_g, h_g\} = 0 \; \forall \; g$ and $\pi_j = 1 \; \forall \; j$ adapting only on ability and ignoring content. The values for π_j could also be configured to manage a-stratification or to seed in field test items alongside operational items as described under the definition of Equation (2).

The most obvious choice for the weights is for them to assume values of 0 or 1 effectively turning the different components of the algorithm off and on, respectively. However, they are real numbers so that they can reflect gradations of importance as decided by test developers. For instance, if $\upsilon_g < \pi_j$, for any particular j and g then more weight is given to the psychometric measurement properties of the items than their content balancing feature. If $\pi_j < \upsilon_g$, then more weight is placed on the content features than the psychometric properties of the items. If $\pi_j = \upsilon_g \; \forall \; j$ and g, then equal emphasis is placed on both. Finally, note that the feature weights are subscripted, indicating they are configurable by content feature giving the algorithm greater preference in some feature areas and less in others.

Hence, the weights are simply inputs to the objective function that are relative to each other. If $\pi_j = 1$ and $\upsilon_g = 1$ equal weight is placed on ability and content in the objective function as it chooses items. Alternatively, the values of $\pi_j = 2$ and $\upsilon_g = .5$ might be used to weight information with greater importance than blueprint features or conversely, $\pi_j = 1$ and $\upsilon_g = 3$ places more weight on feature requirements than on ability in the objective function.

Methods for Adapting on Examinee Ability

The first term in Equation (2) is the item information function used to adapt on examinee ability. This is typically done when items are drawn from a calibrated item pool and the item information function is evaluated at $\widehat{\theta}_{it}$ and is then used to determine the next best candidate item (Chen, Ankenmann, & Chang, 2000; Han, 2009). Achieving this can be accomplished using multiple approaches as noted below.

One of the greater challenges in CAT is that $\widehat{\theta}_{it}$ early in the test is a poor approximation of the true latent trait given that the estimate is based on a small subset of test items (Chen et al., 2000). That is, for initial item selection we effectively have no information regarding the examinee's true ability. However, the asymptotic properties of the estimator, $\widehat{\theta}_{it}$, imply that as more items are administered at each additional step t, the observed estimate approaches its true value, θ_i, and is asymptotically consistent, $\lim_{t \to \infty} \Pr\left(|\widehat{\theta}_{it} - \theta_i|\right) > e = 0$. For this reason, evaluating the item information function at the point estimate $\widehat{\theta}_{it}$ early in the test may prove less than ideal because that point estimate is swamped with measurement error (Chang, 2015).

However, as more items are administered, perhaps the point estimate can be plugged in to compute item information. This suggests a possible switching strategy could be employed to account for the uncertainty in the ability estimate early in the test but then switch to a method that directly uses the point estimate later in the test. Below we describe approaches for computing item information and then describe how switching between them could be an advantage to balance computational overhead with statistical precision.

Maximum Fisher Information

The item information function in item response theory (IRT) for binary items as

$$I_j(\widehat{\theta}_{it}) = D^2 a_j^2 \left[\frac{\left(\Pr_j \left(\widehat{\theta}_{it} \right) - c_j \right)^2}{\left(1 - c_j \right)^2} \right] \left[\frac{1 - \Pr_j \left(\widehat{\theta}_{it} \right)}{\Pr_j \left(\widehat{\theta}_{it} \right)} \right], \tag{3}$$

where $\Pr_j(\widehat{\theta}_{it})$ is the probability of response from a 3-parameter logistic model given the triplet of $\{a_j, b_j, c_j\}$ denoting the IRT parameters of item slope, item location, and item guessing, respectively (Lord, 1980). The maximum Fisher information (MFI) computes item information from $\widehat{\theta}_{it}$ at each step of the algorithm and items can be sorted according to the MFI choosing the item with the largest MFI given $\widehat{\theta}_{it}$ (Chang & Ying, 1996).

Integrating over the Information Function

A second option is to account for the uncertainty in the point estimate and calculate information averaged over the function as

$$I_j(\widetilde{\theta}_{it}) = \int_{\widehat{\theta}_{it} - \delta_t}^{\widehat{\theta}_{it} + \delta_t} D^2 a_j^2 \left[\frac{\left(\Pr_j \left(\theta \right) - c_j \right)^2}{\left(1 - c_j \right)^2} \right] \left[\frac{1 - \Pr_j \left(\theta \right)}{\Pr_j \left(\theta \right)} \right] d\theta. \tag{4}$$

The limits for integration can be formed using confidence intervals from the conditional standard errors, $\sigma(\widehat{\theta}_{it})$, as $\delta_t = \Phi^{-1}(p)\sigma(\widehat{\theta}_{it})$ where $\Phi^{-1}(\cdot)$ is the inverse of the normal cumulative distribution function and p is the nominal coverage probability (Veerkamp & Berger, 1994). Equation (4) is computed using Gauss-Legendre quadrature.

Kullback-Leibler Information

A third approach is via the Kullback-Leibler information method (Cover & Thomas, 1991), who informally describe this as a distance metric between two distributions. Its application in adaptive testing for binary items is

$$KL_j \left(\widehat{\theta}_{it} \right) = \int_{\widehat{\theta}_{it} - \delta_t}^{\widehat{\theta}_{it} + \delta_t} \sum_{l=0}^{1} \log \left[\frac{\Pr \left(x_j = l | \theta \right)}{\Pr \left(x_j = l \middle| \widehat{\theta}_{it} \right)} \right] \Pr \left(x_j = l | \theta \right) d\theta, \tag{5}$$

where x_j is the response to item j and θ is true ability (Belov & Armstrong, 2011; Chang & Ying, 1996; Chen et al., 2000). Chang and Ying (1996) choose $\delta_t = \frac{p}{\sqrt{t}}$ where p is chosen to reflect a nominal coverage probability and justify an impressive implementation relying on the well-documented asymptotic properties of the estimator where the integration limits become smaller with increasing t. Our implementation is similar, but instead of using $\frac{p}{\sqrt{t}}$, we draw the limits of integration using δ_t in the same way described for Equation (4). We again approximate this integral using Gauss-Legendre quadrature.

Partial Adaptive Information Strategy

It is generally true that adaptive algorithms do not require many items to stabilize near the final ability estimate. In many cases, the test is long simply to administer items representing all areas

of the test blueprint, but the ability estimate may have stabilized at some point much earlier in the test. Hence, we might not expect $\widehat{\theta}_{it}$ to change much after some point in the test. If so, one possible consideration is to form provisional values for $\widehat{\theta}_{it}$ only up to step t of the test, where t is some point of stability such as $(\widehat{\theta}_{it} - \widehat{\theta}_{i,t-1}) < \epsilon$ where ϵ is some user-defined tolerance. Then, from step $t+1$ to the end of the test, plug in the last provisional $\widehat{\theta}_{it}$ obtained and from this point forward items are chosen only based on their content requirements fixing information at $\widehat{\theta}_{it}$. Once the examinee completes all required operational responses, generate a final $\widehat{\theta}_{iT}$ using all item responses. This option would save the overhead associated with computing ability estimates and also with item information.

Information Switching Strategy

The KL procedure and the method for averaging over the information function have greater computational expense than directly computing the MFI as they both involve an integral. However, they make better use of the uncertainty in the estimate than the MFI which uses the noisy point estimate. Balancing the two ideas suggests a trade-off approach where one could rely on the asymptotic properties of the estimator and begin early item selection with either the more expensive KL or averaging functions for some initial item selection and then later in the test rely more on the point estimate for computing information.

This idea is originally credited to Chang and Ying (1996). However, our understanding of their approach is that they compute KL over all items, but with increasing values of t, the limits of integration around the observed estimate become smaller, thus treating the observed score as a better estimate at each step. This remains computationally expensive as each step still involves evaluating the integral.

An efficient computational alternative, but theoretically consistent approach, would be to switch to the MFI from either the KL or the average information function after a fixed number of items is administered given the asymptotic expectations for $\widehat{\theta}_{it}$. For example, we might initially compute item information using KL for t steps and then switch to the MFI or, rather than depending on fixed t, switch to the MFI after reaching a desired level of precision.

The notation $I_j(\widehat{\theta}_{it})$, $I_j(\widetilde{\theta}_{it})$, and $KL_j\left(\widehat{\theta}_{it}\right)$ is used to describe different ways to compute information for item j. This is useful to denote the differences in the approach. Going forward, we now generically use $I_j(\widehat{\theta}_{it})$ to describe a method for computing information and describe in narrative which approach is used simply for the convenience of notation.

Content Balancing

The second and third components of the algorithm are related to the content targets to be administered if a test blueprint exists. This component is often critical to support claims of construct validity from an adaptive test. While the term "content balance" is used here, it is perhaps more suitable to refer to them as "feature requirements." These components provide the framework by which expert test developers can appropriately define the construct to be measured and then implementation via the following priority index ensures the test form balances both the psychometric and construct-relevant components simultaneously. Others have proposed managing feature requirements alongside psychometric information within the objective function (Mao & Xin, 2013;

Stocking & Swanson, 1993) and this algorithm aligns with that notion in spirit but differs substantially in computational implementation.

The first content component described here is an index that determines which aspects of the test blueprint should be prioritized at each step while the second component is intended to introduce some randomization such that items have a "boost" to ensure the forms reach their blueprint minimum or maximum expectations.

Content Index

The content balancing *priority index* parameter can be generally defined as in Equation (6), which contributes to the objective function at step t. The term priority index is used to mean that the resulting value contributes to the item selection function adding some priority to improve the chance that item j associated with feature g will be chosen at step t

$$m_{igt} = \begin{cases} 1 + \frac{N_l - n_{igt}}{N_l} & \text{if } n_{igt} < N_l \\ \frac{N_u - n_{igt}}{N_u - N_l} & \text{if } N_l \leq n_{igt} < N_u \\ -p & \text{otherwise.} \end{cases} \tag{6}$$

The values N_l and N_u represent blueprint minimum and maximum values for a particular area of the blueprint. Hence, they are considered boundaries for a particular test feature and are used to mean the range of items on the test form necessary to fulfill that requirement, and the value n_{igt} is the number of items with feature g that have been administered to examinee i up to step t.

The value of m_{igt} initiates at 2 $\forall$ g when $n_{igt} = 0$ and the value approaches unity as $n_{igt} \to N_l$. Once a test feature is satisfied by meeting its minimum requirements (i.e., when $n_{igt} = N_l$), the function of m_{igt} resets to 1 and then the value approaches 0 as $n_{igt} \to N_u$. The parameter includes a penalty once the algorithm achieves the maximum, thus lowering the algorithm's preference to use items with this attribute in future selections. This penalty value p might be $p = 1$ or perhaps any other number. In doing so, the item is not eliminated from potential use, but it adds a negative term to m_{igt}, which will lower its position in the sorted vector $\boldsymbol{\lambda}_{it}$. Hence, when $N_u = n_{igt}$, the chance that such an item would be chosen is minimized and the algorithm will mitigate test blueprint requirements from being exceeded.

How the priority index operates is such that items below their test requirement minimum always have a larger contribution to the objective function than items between their minimum and maximum. That is, items are most attractive to the algorithm when they are below their minimum blueprint requirement and decrease in attractiveness as they approach their maximum.

The function m_{igt} resembles the constraint CAT (CCAT) of Kingsbury and Zara (1989). However, the normalization process described for the rows in the matrix $\boldsymbol{W}$ walks this approach closer in principle to other ways in which content balancing might occur with items containing multiple content features that are not mutually exclusive (Segall & Davey, 1995; Stocking & Swanson, 1993). In this way, if each item in the test bank is a member of one and only one content feature, then $w_{jg} = 1$ and the CCAT is a special case of the approach we implement. If an item belongs to multiple content features, then $0 < w^*_{jg} < 1$ is a normalized value to ensure items that disproportionality share many content features relative to others would not be systematically preferred by the algorithm if these normalized weights were not used.

The allowance for a minimum and maximum creates a blueprint boundary to allow for some variability in the nature of the test form at the subtest level, although an overall test length feature is implemented. For instance, we could enforce a total test length of N but have variability at the subset level. In this way, examinees' i and i' would both have tests of length N. But, they would have different numbers of items at the subtest level making up that total test length. If every test is expected to be exactly the same in terms of length at the subtest level, then $N_l = N_u$ would create tests with different items, but with the same number of items at each lower level.

Content Sequence Balance Parameter

The term m_{igt} in the objective function could cause for the algorithm to be somewhat deterministic and more systematically prefer items with larger values of m_{igt}. To potentially mitigate this effect, the term α_{igt} is introduced as a stochastic parameter to randomly vary the preference of the algorithm for different areas of the content as the test progresses. This parameter adds a small stochastic component to the function at each iteration of the test which has the effect of randomly varying the chances for an item with feature g of being selected at step t.

Some approaches, such as the modified multinomial model (Leung, Chang, & Hau, 2003), have been proposed to deal with the effects of predictable sequencing. Our approach is to add a small stochastic component to the content priority index as $\alpha_{igt} = u_{igt}$ if $\gamma < m_{igt}$ and 0 otherwise where $u_{igt} \sim U(0, 2)$. Note that we draw from $U(0, 2)$ to be commensurate with the values produced by the priority index, m_{igt}. The value set for γ can assume different values with different meaningful consequences. One option is to set $\gamma = 0$, effectively causing the algorithm to randomly prioritize different blueprint features until any given feature reaches its blueprint maximum. A second option is to set $\gamma = 1$, causing the algorithm to accelerate its preference for items associated with a content feature below its blueprint minimum. In this way, it offers some assurance that content feature minimums are met with higher priority than feature maximums. A third option is to set $\gamma = u_{igt}$. In this variant, as m_{igt} tends to 0, the probability that u_{igt} adds to the priority index also decreases given that $\Pr(u_{igt} < m_{igt}) = \frac{m_{igt}}{2}$ from properties of the uniform cumulative distribution function. In this way, the algorithm will tend to boost values for features with larger values of m_{igt}.

Feature Scaling

The item information function and content priority index in Equation (1) are unique metrics each on a different scale. Hence, summation over those terms without any normalization would cause for one metric to have a larger influence than another simply as a function of the different metric space. As such, both metrics are normalized using feature scaling resulting in a continuous number in the $[0, 1]$ space for each. We use min-max feature scaling for each term and the information function, for example, is rescaled as $I_j'(\widehat{\theta}_{it}) = \frac{I_j(\widehat{\theta}_{it}) - I(\widehat{\theta}_{it,\min})}{I(\widehat{\theta}_{it,\max}) - I(\widehat{\theta}_{it,\min})}$ where $I(\widehat{\theta}_{it,\max})$ and $I(\widehat{\theta}_{it,\min})$ denote the maximum and minimum item information functions of all items evaluated for examinee i at step t, respectively. This rescaling is similarly applied to m_{igt}. Without this rescaling, the difference in the metrics between $I_j(\widehat{\theta}_{it})$ and m_{igt} would arbitrarily cause for one of them to play a larger role in the objective function. Note that if $\sum_{g=1}^{G} \upsilon_g = 0$, (i.e., all blueprint weights are 0) then feature rescaling is not performed on the content index. In this case, the denominator in the rescaling

would be 0 and min/max feature scaling would be undefined.

Exposure Control

Item exposure control is perhaps one of the most widely studied topics in CAT. Here, we limit consideration to only two methods for implementation in our work, but other methods could easily be incorporated as decision variables for building a CAT engine. The first approach requires no prior knowledge about the items or the population of examinees and is simple to implement. The second is an IRT-based approximation to prevent items with large a-parameters from being selected too often. We acknowledge that the Sympson-Hetter method (Sympson & Hetter, 1985) is an important exposure control method in the CAT literature. Its implementation requires some work to establish an exposure control parameter whereas the randomesque requires no prior knowledge or exposure limits on items. For this reason, we focus on the two methods below, but the Sympson-Hetter method could very easily fit into our algorithmic framework.

Randomesque Approach

The randomesque approach is implemented by computing the value of λ_{jit} for all items in the bank at the current, provisional, value of $\widehat{\theta}$ which forms a vector, sorted in descending order, $\boldsymbol{\lambda}_{it} = \{\lambda_{1it},\ \lambda_{2it,} \ldots, \lambda_{Nit}\}'$ where N is the number of (remaining) items in the bank. In this way, the first item in the vector provides the most information and the final item in the vector provides the least. Implementation begins by setting a configurable parameter $w = \{1, 2, 3, \ldots, N\}$ and randomly choosing one item from $\{\lambda_{1it},\ \lambda_{2it,} \ldots, \lambda_{wit}\}$. If $w = 1$, the algorithm always chooses the top, best item, and if $w = N$ the algorithm is effectively taking a random draw from all available items in the bank. Pragmatically, setting $w = 5$ (or approximately so) ensures some balance between randomization and selection of the best next item (Georgiadou, Triantafillou, & Economides, 2007; Kingsbury & Zara, 1989). This randomization also adds a security feature for examinees in similar proctored settings. This approach mitigates the potential for individuals in similar settings to engage in answer copying since items are partially random even for those with similar ability estimates.

The Impact of Item Slopes

All things being equal, items with large a-parameters provide the most psychometric information conditional on θ and algorithms using IRT-based information functions are likely to overexpose those items (Hau & Chang, 2001). Some approaches consider stratification methods to form bins such that items with smaller slopes are administered earlier in the test and items with larger slopes would be administered later (Huebner et al., 2018). Our algorithm could manage a-stratification using the reciprocal of the a-parameters as weights as described under Equation (2). Here we propose a simpler method to minimize the exposure risk of items with large a-parameters by provisionally setting $a_j = a \,\forall\, j$ as a common pseudo-value only when computing the item information for purposes of item selection. That is, we use this approximation in item selection but use the estimated (actual) a-parameter when generating ability estimates, both provisional and final. This

approximation on the a-parameter is one additional exposure control parameter that mitigates the overexposure of highly informative items in certain populations with similar abilities.

Provisional Scoring Options

A CAT progressively updates $\widehat{\theta}_{it}$ after a response is submitted to each new item (Wang & Vispoel, 1998). With IRT, this is generally based on concepts of maximum likelihood estimation (MLE) or Bayesian methods (Bock & Mislevy, 1982; Spray & Reckase, 1996), with the latter being preferred. The challenge with MLE scoring in CAT is that ability estimates cannot be provided for response strings with perfect answers or completely wrong answers as the likelihood function has no maximum. Some tricks can be implemented for MLE methods to adjust score strings or to supplement the response with a fake item response, sometimes referred to as "fences" (Han, 2016). Instead, Bayesian methods do not suffer from this same problem and can be used to generate ability estimates in a broader case of scenarios.

Termination Criteria

Termination of the algorithm may occur under multiple conditions (Babcock & Weiss, 2009), three of which are noted here. The first option is to stop when $\sigma(\widehat{\theta}_{it}) \leq \varepsilon$ where ε is a user-defined tolerance for stopping. A second option is when the provisional ability stabilizes as in the partial adaptive strategy, $(\widehat{\theta}_{it} - \widehat{\theta}_{i,t-1}) \leq \varepsilon$. Or a third option is to stop once the algorithm has administered a fixed number of items. The first two are related to the measurement precision and the third option may be used when feature requirements need to be satisfied before stopping.

Item Selection Process

With all individual components of the algorithm now described, we can provide a high-level schema for item selection as a set of steps as shown in Algorithm (1).

Algorithm 1 Procedural Schema for CAT Algorithm

Input: Choose initial item j from bank and administer to examinee i at step $t = 0$.

1: Form a provisional ability estimate after the examinee responds, $\widehat{\theta}_{it}$.
2: Update m_{igt} based on item administered at step t and then set $t := t + 1$.
3: Recompute $I_j'(\widehat{\theta}_{it})$ given new estimate, $\widehat{\theta}_{it}$.
4: Compute $\boldsymbol{\lambda}_{it}$ using Equation (2) for the remaining items in the bank excluding previously administered items.
5: Sort vector $\boldsymbol{\lambda}_{it}$ in descending order.
6: Apply an exposure control procedure and select one item from $\boldsymbol{\lambda}_{it}$.
7: If termination criteria are satisfied, stop. Else, return to (1).
8: After test termination, generate final $\widehat{\theta}_{iT}$ for reporting.

Considerations for Algorithm Scalability

Choi et al (2016) explore an improved computation of the shadow test and make the claim that cloud-based infrastructures for computing are "scalable without limit." This is true in the sense that cloud-based structures provide virtually limitless access to fast computers. This is untrue in the sense that it comes with increasing monetary cost and when building operational environments for industry, that monetary cost indeed has limits. Consequently, it is important to find the best algorithm for computing and only then pass that into larger cloud-based systems.

The psychometric literature commonly reports simulation runtimes to justify an algorithm or particular computational approach. While there is some value in these timings, we do not follow this practice here. First, these reported runtimes are idiosyncratic to the machine used in the simulation and only reflect that implementation, limiting its potential generalizability. More importantly, time complexity using Big $\mathcal{O}$ is the *de facto* measure by which an algorithm's ability to scale is evaluated (Chivers & Sleightholme, 2015). MILP algorithms have a time complexity of $n^n 2^{\mathcal{O}(n)}$ (Dadush, Eisenbrand, & Rothvoss, 2023) but quicksort algorithms have complexity $\mathcal{O}(n\log(n))$ (Shujaat, 2022), letting n, in this case, represent the collection of inputs used for the model. Hence, all things being equal, quicksort will scale better than MILP. This does not guarantee our approach will have a faster runtime than MILP as timing depends on implementation. It does mean, that all things being equal, our algorithm will scale better and is more likely to have a faster runtime with comparable implementation strategies. On balance, MILP solvers are substantially improving (Koch, Berthold, Pedersen, & Vanaret, 2022) but the runtimes for even the most popular solvers such as CPLEX remain an open question (IBM, 2023).

It is also important to state that any runtimes observed in simulation settings are well-controlled environments. Real-world servers are hit with large numbers of simultaneous users in live testing in unpredictable ways. Cloud computing structures manage large traffic and computing volumes via "elastic load balancing", which dynamically causes additional server instances to be used when placed under high demand. This would permit any number of CAT algorithms to seem fast, but at the additional monetary cost associated with additional servers need to support the larger computational demands.

Equation (1) simultaneously computes the information function and the content priority index at each step over all items in the bank for each examinee to provide the overall value of λ_{jit}. It is already an efficient implementation given that we treat the inputs as fixed values and the values for λ_{jit} are directly realized. However, computing information for all J items in Equation (1) has some computational cost and the sorting algorithm itself has cost. Both of those issues can be addressed to reduce the search space and improve computational speed. As a brief aside, one fundamental difference between our approach and the shadow test is our computational cost is limited to the selection of only the next item needed to satisfy constraints. The shadow test builds an entire form of items at each step that must satisfy all constraints but then chooses only one item from that form at each step.

We begin with the notion that realizing scalability is not a function of bigger computing environments, per se. The first objective is to analyze the complexity of the algorithm and then make use of distributed computing concepts or cloud-based structures for faster computing. Hence, we could make use of two common ideas. One could be to precompute values of information for all items at fixed increments of θ and use a precompute, store, and recall approach to create a lookup table. This could be useful, but given that $\theta \in \mathbb{R}$, these values would never be exact for the exam-

inee. Another option is to use parallel processing and distribute the computing of information for J items over multiple cores. This has significant potential, but the split, apply, combine problem brings with it some overhead of reassembling components and this should be evaluated (Wickham, 2011).

Because the algorithm's runtime is proportional to its inputs—specifically computing over the number of items in the test bank, reducing the number of computations over the space of J reduces overhead substantially (Doran, 2023). Here we describe an option to reduce the number of inputs for computational scaling.

A Feasible Computational Approximation with Reduced Inputs

Our proposed computational option is to compute λ_{jit} in smaller components instead of simultaneously over the entire search space. Those steps could be outlined as two steps. First, compute the value of the function based initially on the content features over all items in the bank $\lambda^1_{jit} = \sum_{g=1}^{G} \upsilon_g w_{jg} m_{igt} + \sum_{g=1}^{G} h_g w_{jg} \alpha_{igt}$. Following this step, sort the items with the configurable parameter $s = \{1, 2, \ldots, S\}$ as $\lambda^1_{jit} = \{\lambda^1_{1it}, \lambda^1_{2it}, \ldots, \lambda^1_{sit}\}$ and then compute the information value only for the items in this small subset $\lambda^2_{jit} = \pi_j I^{'}_j(\widehat{\theta}_{it}) + \lambda^1_{jit} \; \forall \; j \in \{1, \ldots, s\}$. Then, update the sorted values based on λ^2_{jit} and implement an exposure control approach to choose one item from the smaller set in the new space.

This approximation significantly reduces the computational overhead relative to the more costly component (i.e., the information function) by reducing the space over which it is computed. Specifically, we now compute item information only for s items at each step rather than over all J items, and when $s \ll J$ the cost savings is substantial. Additionally, the sort algorithm searches over a smaller space. Consequently, the idea realizes two computational cost savings. However, this approach implicitly places greater emphasis on the content features of the test than the measurement properties. There could be items with larger values for information outside the set of λ^2_{jit} that would be ignored with smaller values for s.

If the content features are not used by the algorithm for choosing items, an approximation to reduce the search space only based on the item measurement properties could be to initially compute the item information for all items in the bank. Then, after t steps, use an approximate b-matching procedure (Han, 2018) to compute the item information function only for the subset of items in some range of the provisional ability estimate. The b-matching process is equivalent to using the item information with the Rasch model, but with more general IRT models this would only be an approximation to reduce the search space.

To illustrate, consider that an initial ability estimate is obtained after administering t items, $\widehat{\theta}_{it}$. Now, because items and examinees are on the same scale, locate items in the bank where $b_j \in \{\widehat{\theta}_{itl}, \ldots, \widehat{\theta}_{itu}\}$ where the terms l and u are used to mean some lower and upper boundaries around the provisional ability. Then, compute information only for items within this space for the next item selection. This technique also implicitly assumes a common a-parameter and would also help with exposure control.

It could be possible to even further speed up this two-step process using the partial adaptive information strategy suggested. In Step (1) update λ^1_{jit} as described above. Then, in Step (2), recall the stored values of $I^{'}_j(\widehat{\theta}_{it})$ and use those to update λ^2_{jit}. This avoids having to recompute $\widehat{\theta}_{it}$ and $I^{'}_j(\widehat{\theta}_{it})$ and it would be possible to immediately determine all remaining items to be administered

given that we know what aspects of the blueprint need to be satisfied.

Demonstrating the Model Under Variable Conditions

Some behaviors of the algorithm using a simulated item bank and blueprint are provided in this section. Because the algorithm is designed to be highly generalizable to various testing conditions, we can only provide a few reasonable permutations to explore. For this reason, we have also provided a complete testing environment for this algorithm in a web-based tool where all simulations and ideas expressed in this manuscript can be independently verified or users can test their own unique item banks, blueprints, and requirements to further explore the ideas discussed. The web-based application is available at `https://catsim.shinyapps.io/humrro_cat_engine/`.

Simulated Data

The following simulations are based on an item bank of 1000 items from a 2-parameter logistic model items with a-parameters generated from $a_j \sim U[.5, 1.5]$ and $b_j \sim U[-4, 4]$. Items are equally assigned to one of four content features and a blueprint (BP) with minimum (N_l) and maximum (N_u) values as in Equation (6) is created as provided in Table (2). This yields 250 items per dimension with item parameters uniformly distributed within each dimension.

Dimension	Minimum	Maximum
Dimension 1	5	15
Dimension 2	5	20
Dimension 3	5	10
Dimension 4	5	15

Table 2: Blueprint for Simulation

The simulation generates 100 replicates at each true value of θ_q between $q = \{-3, -2.5, \ldots, 3\}$ in increments of .5. Hence, the total number of replicates is 1,300. Using this uniform distribution permits for us to capture and report the root mean squared error and the bias at each θ_q. The observed response to each item is generated by first computing the true probability of response, $\Pr(\theta_q) = (1 + \exp[a_j(\theta_q - b_j)])^{-1}$, and then computing an observed binary response, x_{ijt}, for examinee i to item j at step t where $x_{ijt} = 1$ if $\Pr(\theta_q) < u_{ijt}$ and 0 otherwise where $u_{ijt} \sim U[0, 1]$.

The fixed length simulations using the content blueprint are run with a test of length 50 drawing from the various blueprint dimensions. The statement "100% blueprint match" is used to mean every one of the simulated tests has 50 total items and has item counts within the minimum and maximum boundaries for each dimension of the blueprint. Any deviation from this expectation is considered to not meet the blueprint. In all simulations, we compute the EAP as the provisional estimate and the final score is the MLE.

Exploring Different User-Defined Configuration Weights

We begin with a few logic tests to evaluate whether the algorithm can behave as a generally adaptive item selection algorithm consistent with adaptive theory. Specifically, we explore four conditions and capture a few measures of adaptivity. The first measure will be the root mean squared error (RMSE) at each true value of θ_q. The second will be a ratio of test information functions (TIFs) compared to a theoretical maximum (Wyse & McBride, 2021). The theoretical maximum is determined by computing item information at each true value θ_q and taking their sum over the 50 most informative items at each θ_q. This is only a "theoretical" maximum in the sense of what this specific simulated bank could support.

The first condition will be a test that chooses items from the bank entirely randomly. This serves as a general baseline condition that has no adaptiveness to either the examinee or the test blueprint. Hence, under this condition, we would expect large RMSE relative to adaptive conditions and we expect the match to blueprint at rates only equal to random chance, which will be effectively 0%. Recall that our definition of "meeting blueprint" is one where the total test length is 50 and the number of items by dimension is within the min/max boundaries. Achieving this randomly would have a probability roughly equal to $(n_g/250)^4$ where n_g is the number of items within each dimension, although n_g can vary by dimension. The second condition is a linear-on-the-fly (LOFT) variant that adheres to a content blueprint. In this condition, we would expect the algorithm to choose items based on their content features and return a blueprint match rate of 100%. Because there is no adapting to examinee ability, LOFT chooses items randomly within each dimension as a stratified random sampling approach, and so we expect the RMSE to be comparable in magnitude to the random condition. The third condition is purely item-adaptive and ignores content requirements. In this condition, we expect match to blueprint rates equal to what would occur under random chance and very small RMSE relative to non-adaptive conditions as items are now selected based on examinee ability. Our fourth condition is both item-adaptive and content-adaptive. Here we expect a 100% match to blueprint and, like the item adaptive condition, we expect RMSEs that are comparable in size to the item adaptive condition. These measures and the logic are similar to those used by van der Linden (2022).

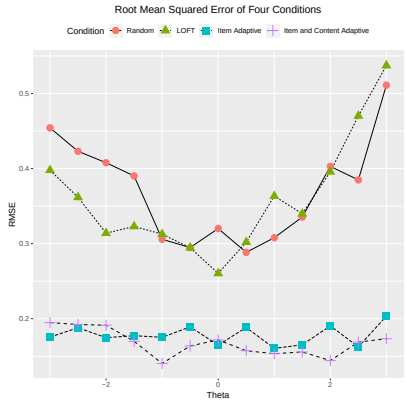
Table (3) describes the different weight configurations used to yield each result. Additionally, the table shows the observed blueprint match rate from each of the simulated conditions. Figure (1a) shows the RMSE for each tested condition at the increments of θ_q used in the simulation and Figure (1b) shows the ratio of the TIFs. Results in both figures match *a priori* expectations for the RMSE and TIF ratios such that RMSEs for item-adaptive conditions are smaller than non-adaptive and the TIF ratios approach 100% for the adaptive conditions. Both suggest that the item selection algorithm described in Equation (1) achieves its two primary objectives. First, it is appropriately adapting on the examinee ability in Conditions (3) and (4) as observed in the smaller RMSE and larger TIF ratios. Second, small changes to the configuration weights achieve very different testing conditions and the match to blueprint rates indicate the content priority index achieves its objectives to adhere to a content blueprint.

The Potential of Information Switching

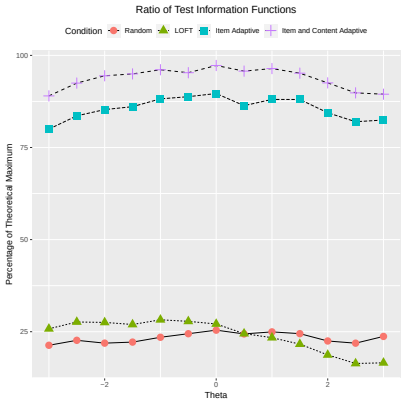
A second question explored is whether the notion of switching information after t steps offers benefits. In this portion of the simulation, we implement a stopping criterion using the conditional

Configuration Weight	Condition	Observed BP Match Rate
$\pi = 0, \upsilon = 0, h = 0$	Random Condition (Condition 1)	0%
$\pi = 0, \upsilon = 1, h = 1$	Linear on the Fly (LOFT) (Condition 2)	100%
$\pi = 1, \upsilon = 0, h = 0$	Item Adaptive (Condition 3)	0%
$\pi = 1, \upsilon = 1, h = 1$	Item and Content Adaptive (Condition 4)	100%

Table 3: User-Defined Configuration Weights for Simulation



(a) *RMSE of Four Simulated Conditions.*



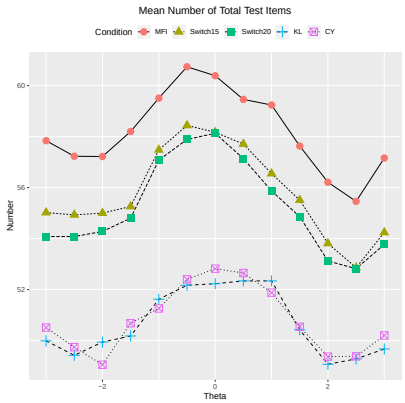
(b) *Percentage of Theoretical Maximum.*

Figure 1: Two Measures of Adaptivity

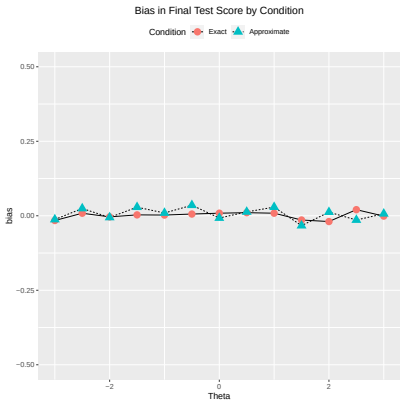
standard errors of measurement, $\sigma(\widehat{\theta})$, such that each test terminates once $\sigma(\widehat{\theta}_{it}) < .2$. Here five conditions are explored to assess this question. Condition 1 (MFI) uses the MFI for computing item information for all items administered. Condition 2 (Switch15) uses the KL procedure for the first 15 items and then switches to the MFI. Condition 3 (Switch20) uses the KL procedure for the first 20 items and then switches to the MFI. Condition 4 (CY) computes the KL information demonstrated by Chang and Ying (1996) where we use $d = .95$ as the coverage probability. Last, Condition 5 (KL) uses the KL procedure for all items and always uses a one standard error boundary around the provisional ability estimate. This concept has been explored to assess recovery on the estimated latent trait (Chang & Ying, 1996), but here we explore its impact on the test length.

Both Conditions 1 and 5 intend to provide upper and lower bound limits in terms of what we might expect in terms of total test length, given the limits of this specific item bank. If Conditions 2 or 3 offer any benefit, we expect to observe that the test would reach its termination criteria with a fewer number of test items relative to Condition 1 and approach the test length observed in Condition 5. Figure (2a) shows the mean total number of test items for each of the conditions over values of θ_q. Here we observe that over the entire range of ability, Condition 2 reaches the termination criteria with fewer test items relative to Condition 1, typically between 2 to 3 fewer items, effectively reducing test length by about 4% relative to Condition 1. Additionally, we observe Condition 3 even further reduces test length beyond Condition 2 by about one more test item. It is useful to note that the CY condition seems on par with the KL procedure even though it tends to rely on the observed score more as the test increases in length.

These simulations suggest the idea of switching how information is computed helps to reduce test length by capturing the uncertainty in the ability estimate early in the test. We almost certainly



(a) *Mean Total Number of Test Items by Condition.*



(b) *Observed Bias by Condition.*

Figure 2: Impact of Switching and Pseudo Item Slope Impact on Bias

do not want to compute the KL for all items as this is computationally expensive relative to the MFI, although doing so seems to provide the shortest test length. Hence, by choosing a value of t where switching occurs, we approach the lower bound result of Condition 5 while reducing computational overhead as we switch to the MFI. This is a promising result for test developers wishing to even further reduce test length.

Exposure Control with Approximate Item Slopes

Here we test the idea of fixing the a-parameter to a pseudo-value for purposes of choosing items, but use its actual estimate when generating scores, both provisional and final. We explore two conditions. In Condition 1, the estimated (actual) a-parameter is used and we measure the proportion of the bank that is used across all replicates in the simulation. In Condition 2, we fix $a_j = 1 \; \forall \; j$ for item selection and the proportion of the bank is again measured. Additionally, the final test score is captured as the MLE under both conditions and the bias between both conditions is considered.

Figure (2b) illustrates the bias at all true values of θ_q between both conditions. This figure shows that both conditions have no real practical differences in bias indicating the approximate a-parameter does not tend to compromise the final test score negatively. Additionally, and importantly, the approximate a-parameter tends to use a larger proportion of the item bank. In Condition 1, the simulation shows the exact a-parameter results in the usage of 83% of the bank and the approximate a-parameter results in the usage of 96.4% of the bank.

Item Enemy Administration

Item enemies are sets (or pairs) of items that cannot be administered to an examinee on the same test form. Constructing fixed forms to avoid item enemies is straightforward as known pairs of enemies can be intentionally avoided in form construction. Here we demonstrate that the proposed algorithm can handle item enemy detection as a feature requirement without needing to create a bifurcated item bank. While we demonstrate this using item enemies, really the intent is to show that the feature requirements can be any arbitrary grouping structure defined by test developers. The potential here is virtually unlimited.

The example is constructed by forming a cluster of ten items that are enemies with each other. Practically speaking, this means if one of these ten items is administered on a test form, then none of the remaining nine can appear on the same test form because they are enemies. We do this by forming a binary indicator in the item metadata associated with each item in the bank denoting a 1 if the items are in the enemy cluster group and 0 otherwise. Second, we modify the original test blueprint in Table (2) to have two new feature requirements. We add Group 1 with a min/max of 0/1 and Group 2 with a min/max of 49/50. The min/max of 0 to 1 for Group 1 instructs the algorithm to choose no more than one item from within this "enemy" cluster, but it can optionally choose none and Group 2 instructs the algorithm where to choose the other 49 (or 50) items.

We again simulate this new feature requirement which results in a 100% match to blueprint over all replicates and also no test form contains more than one item from Group 1 as expected by our new structure.

A Note on Items Crossed with Multiple Features

The modified blueprint used for the enemy example provides insight into how the algorithm can behave in complex ways. The term "Enemy Group" is used only as a descriptor for the example as those could be any conceivable grouping. The Group 1 min/max in that example is set to 0 and 1, respectively, to ensure no more than one item in Group 1 is administered together. However, a maximum larger than 1 would cause multiple items from within this group to be administered. Here we explain how the algorithm extends from nested features to scenarios where items are linked to multiple features.

First, when items are linked to one and only one feature, we say they are nested and the algorithm freely chooses n_g items from within each feature until test requirements are satisfied. For example, referencing Equation (1) and considering the nested simulation examples with four mutually exclusive dimensions, there would be four values of m_g, one for each dimension. So, the feature-relevant portion for the jth item (in simplified scalar form and ignoring item information for now) is $w_{j,d1}m_{d1} + w_{j,d2}m_{d2} + w_{j,d3}m_{d3} + w_{j,d4}m_{d4}$. Recall that larger values of m_g as defined by Equation (6) cause for items linked to feature g to have greater appeal to the item selection algorithm and when a feature maximum has been realized, then $m_g = -p$. For example, suppose Dimension 4 has reached its max but the others have not. Then $m_4 = -p$ and items in Dimensions 1 to 3 would have positive values for m_g and be sorted higher in $\boldsymbol{\lambda}_{it}$ than any item associated with Dimension 4. We also remind the reader that w_j is used when items are linked to a single feature and w_j^* is used to denote a normalized value for items linked to multiple features.

In more complex scenarios, items can be linked with multiple features and not simply nested. For example, an item can simultaneously be linked with Dimension 1 and also Group 1 or Group 2. As such, there would now be six values of m_g, one for each of the four Dimensions and the two for Groups 1 and 2. So, the feature-relevant portion of the equation is $w^*_{j,d1}m_{d1} + w^*_{j,d2}m_{d2} + w^*_{j,d3}m_{d3} + w^*_{j,d4}m_{d4} + w^*_{j,g1}m_{g1} + w^*_{j,g2}m_{g2}$. Once Group 1 reaches its maximum, then $m_{g1} = -p$ and a negative term is added to items linked with this feature. However, if Group 2 has not reached its maximum, m_{g2} is a positive term added to items linked with this feature. For example, $w^*_{j,d1}m_{d1} - w^*_{j,g1}p$ is the value for the jth item linked to Dimension 1/Group 1 when Group 1 has reached its max and $w^*_{j,d1}m_{d1} + w^*_{j,g2}m_{g2}$ is the value for the jth item in Dimension 1/Group 2 when Group 2 has yet to reach its max. This is where the value of the penalty term, p, is used so that $w^*_{j,d1}m_{d1} - w^*_{j,g1}p < w^*_{j,d1}m_{d1} + w^*_{j,g2}m_{g2}$, thus causing items associated with Dimension1/Group

2 to be sorted higher in the vector $\boldsymbol{\lambda}_{it}$ than those associated with Dimension 1/Group 1.

The addition of the min/max for Group 2 is not critical in this specific example as $w^*_{j,d1}m_{d1} - w^*_{j,g1}p < w^*_{j,d1}m_{d1}$ would be true causing the algorithm to prefer items in Dimension 1 that are not in Group 1. It is used here only to complete the example where there could be other min/max conditions for many more than two groups where this would be needed.

On-the-Fly Multistage

The simulations showcase the item-adaptive and the LOFT variants. We can creatively see the generalized nature of Equation (1) can also support on-the-fly multistage adaptive tests (Zheng & Chang, 2015), which sits in between these two ideas. Suppose the goal is to build smaller content-balanced forms that maximize information near a classification cutpoint. To form an example, let there be two ordered cutscores for classification, $\theta_1 < \theta_2$. We first fix information at θ_1, then, all that remains is to choose items based on their content features like what is demonstrated for LOFT. A similar process would occur for θ_2 so that there is now an "easy" form and a "hard" form. In this way, we're effectively building on-the-fly adaptive forms targeting a specific level of ability and examinees can be routed to other similarly built forms based on their scores after completing an initial form.

Discussion

Our effort here has been to describe and demonstrate thorough details of a new generalizable item adaptive algorithm. The formalities of the algorithmic components are defined alongside various decision details that could be used for its implementation. For completeness of presentation, we provide computational details so this work can stand alone as a specification resource for others wishing to implement Equation (2). Additionally, we demonstrate the behavior of the algorithm using simulated conditions in various ways. These simulation results illustrate its behavior is consistent with the stated expectations and can achieve an array of custom test designs by simply changing the user-defined importance weights and configuring the priority index to achieve test developer-defined outcomes. The examples demonstrated here represent only a small subset of what this algorithm can achieve.

While some of the mathematical formalities may initially seem formidable, it is quite simple to implement and it has significant potential to scale with large item banks, many test requirements, and large volumes of simultaneous test takers. Specifically, Equation (2) can be implemented directly or the alternative variant that reduces the search space can be used. The emphasis placed on scaling the algorithm is intended to address real-world applications of this algorithm's potential in two specific ways. First, faster identification of the items needed to fulfill test requirements minimizes observed latency at the test-taker level. Not only does latency disrupt an examinee's experience, but the vendor providing the test platform may also be subject to financial damages if they occur, what is commonly termed "liquidated damages". Second, this algorithm scales in a way that is more efficient than other proposed CAT approaches. This means if pushed into cloud-based structures for computing, it is likely to require fewer servers on demand, thus reducing monetary cost.

While we purport that the algorithm is highly configurable and useful for many different testing situations, we note that these claims are based on our experiences and the typical uses cited in the literature. The world of test designs and optimization approaches with machine learning technologies is rapidly changing and our claims are based on practices commonly used at this time. Additionally, the ideas explored here are based on unidimensional IRT models. While potentially a limit, multidimensional IRT to date has not become common in large-scale assessment and unidimensional IRT models continue to be the most common in real-world applications.

Going forward, a few ideas can be further explored. One idea to consider is whether optimization can be performed with complexity better than $\mathcal{O}(n \log n)$. Our description above requires sorting only because we use the randomesque approach. If instead we simply were to choose $\max \boldsymbol{\lambda}_{it}$, then the algorithm would scale as $\mathcal{O}(n)$. The evaluation of algorithms is often missing from psychometric conversations, but with cloud-based systems now offering platforms for psychometric computing, such an evaluation is critical for cost containment. Cloud-based computing platforms indeed offer tremendous power and speed. However, this benefit comes with real monetary cost so we recommend that the first goal be to find the more optimal algorithm and only then pass the algorithm into a bigger computing environment. A second idea to explore is if other ways to compute item information could be used in the information switching idea to reduce test length even further. The idea we demonstrated has promise, and with concerns over testing time being generally too long, this could help reduce that concern. In particular, the newer notion of "through-year" designs in education could benefit from shorter, more precise tests, to provide diagnostic information at the examinee level. In fact, the State of Virginia in the U.S. passed legislation in 2021 requiring through-year designs with a CAT (Virginia State Statute §22.1-253.13:3, 2021).

It will be interesting to speculate on future CAT algorithms combined with automated item generation (AIG), item forecasting, and personalized learning platforms. Imagine an examinee sitting with a device that uses AIG to generate an item with forecasted item parameters, administer the item, collect and score the examinee's response, repeat AIG to develop a new item based on the examinee's response, and iterate this process until completion of the test. Then serve tailored, personalized learning content based on how the examinee performed on the test. Future integrations of large language models within psychometric platforms have tremendous promise.

References

Babcock, B., & Weiss, D. J. (2009, June). *Termination criteria in computerized adaptive tests: Variable-length cats are not biased* (Tech. Rep.). Presented at the Realities of CAT Paper Session.

Belov, D., & Armstrong, R. (2011, 05). Distributions of the Kullback-Leibler divergence with applications. *The British journal of mathematical and statistical psychology*, *64*, 291-309.

Belov, D., Armstrong, R., & Weissman, A. (2008, 09). A Monte Carlo approach for adaptive testing with content constraints. *Applied Psychological Measurement*, *32*, 431-446.

Bock, R. D., & Mislevy, R. J. (1982). Adaptive EAP estimation of ability in a microcomputer environment. *Applied Psychological Measurement*, *6*(4), 431-444.

Chang, H.-H. (2015). Psychometrics behind computerized adaptive testing. *Psychometrika*, *80*(1), 1-20.

Chang, H.-H., & Ying, Z. (1996). A global information approach to computerized adaptive testing. *Applied Psychological Measurement*, *20*(3), 213-229.

Chen, S.-Y., Ankenmann, R., & Chang, h.-h. (2000, 09). A comparison of item selection rules at the early stages of computerized adaptive testing. *Applied Psychological Measurement*, *24*, 241-255.

Chivers, I., & Sleightholme, J. (2015). An introduction to algorithms and the Big O notation. In *Introduction to programming with Fortran: With coverage of Fortran 90, 95, 2003, 2008 and 77* (pp. 359–364). Cham: Springer International Publishing.

Choi, S. W., Moellering, K. T., Li, J., & van der Linden, W. J. (2016). Optimal reassembly of shadow tests in cat. *Applied Psychological Measurement*, *40*(7), 469-485.

Cohen, J., & Albright, L. (2014). *Smarter balanced adaptive item selection algorithm design report.* (Tech. Rep.). Retrieved from `http://www.smarterapp.org/specs/AdaptiveAlgorithm.html`

Cover, T. M., & Thomas, J. A. (1991). *Elements of information theory*. New York: Wiley.

Dadush, D., Eisenbrand, F., & Rothvoss, T. (2023). From approximate to exact integer programming. In A. Del Pia & V. Kaibel (Eds.), *Integer programming and combinatorial optimization* (pp. 100–114). Cham: Springer International Publishing.

DMDC. (2008). *CAT-ASVAB Forms 5-9 (technical bulletin no. 3)* (Tech. Rep.). Defense Manpower Data Center. Retrieved from `https://www.officialasvab.com/wp-content/uploads/2019/08/asvab_techbulletin_3.pdf`

Doran, H. (2023). A collection of numerical recipes useful for building scalable psychometric applications. *Journal of Educational and Behavioral Statistics*, *48*(1), 37-69.

Florida State Statute §1008.25. (2023).

Georgiadou, E. G., Triantafillou, E., & Economides, A. (2007). A review of item exposure control strategies for computerized adaptive testing developed from 1983 to 2005. *The Journal of Technology, Learning and Assessment*, *8*(5), 431-446.

Han, K. C. T. (2009). *Gradual maximum information ratio approach to item selection in computerized adaptive testing* (Tech. Rep.). Retrieved from `https://www.gmac.com/market-intelligence-and-research/research-library/validity-and-testing/research-reports-validity-related/gradual-maximum-information-ratio-approach`

Han, K. C. T. (2016, 05). Maximum likelihood score estimation method with fences for short-length tests and computerized adaptive tests. *Applied Psychological Measurement*, *40*, 289-301.

Han, K. C. T. (2018). Components of the item selection algorithm in computerized adaptive testing. *Journal of educational evaluation for health professions*, *15*(7).

Hau, K., & Chang, H. (2001). Item selection in computerized adaptive testing: Should more discriminating be used first? *Journal of Educational Measurement*, *38*(3), 249–266.

Ho, T.-H., & Dodd, B. G. (2012). Item selection and ability estimation procedures for a mixed-format adaptive test. *Applied Measurement in Education*, *25*(4), 305-326.

Huebner, A., Wang, C., Daly, B., & Pinkelman, C. (2018). A continuous a-stratification index for item exposure control in computerized adaptive testing. *The British journal of mathematical and statistical psychology*, *42*(7), 523-537.

Huebner, A., Wang, C., Quinlan, K., & Seubert, L. (2015, 10). Item exposure control for multi-dimensional computer adaptive testing under maximum likelihood and expected a posteriori estimation. *Behavior research methods*, *48*.

IBM. (2023). *Problem characteristics affecting CPLEX run time.* Retrieved from `https://www.ibm.com/support/pages/problem-characteristics-affecting-cplex-run-time`

Kingsbury, G. G., & Zara, A. R. (1989). Procedures for selecting items for computerized adaptive tests. *Applied Measurement in Education*, *2*(4), 359-375.

Koch, T., Berthold, T., Pedersen, J., & Vanaret, C. (2022). Progress in mathematical programming solvers from 2001 to 2020. *EURO Journal on Computational Optimization*, *10*.

Leung, C.-K., Chang, H.-H., & Hau, K.-T. (2002). Item selection in computerized adaptive testing: Improving the a-stratified design with the sympson-hetter algorithm. *Applied Psychological Measurement*, *26*(4), 376-392.

Leung, C.-K., Chang, H.-H., & Hau, K.-T. (2003). Computerized adaptive testing: A comparison of three content balancing methods. *The Journal of Technology, Learning and Assessment*, *2*(5).

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

Luecht, R. M., De Champlain, A., & Nungester, R. J. (1997). Maintaining content validity in computerized adaptive testing. In A. J. J. A. Scherpbier, C. P. M. van der Vleuten, J. J. Rethans, & A. F. W. van der Steeg (Eds.), *Advances in medical education* (pp. 366–369). Dordrecht: Springer Netherlands.

Ma, H., Zeng, Y., Yang, S., Qin, C., Zhang, X., & Zhang, L. (2023). A novel computerized adaptive testing framework with decoupled learning selector. *Complex and Intelligent Systems*, *9*, 5555-5566. Retrieved from `https://doi.org/10.1007/s40747-023-01019-1`

Mao, X., & Xin, T. (2013). The application of the Monte Carlo approach to cognitive diagnostic computerized adaptive testing with content constraints. *Applied Psychological Measurement*, *37*(6), 482-496.

Mislevy, R. J., & Riconscente, M. M. (2006). Evidence-centered assessment design. In S. M. Downing & T. M. Haladyna (Eds.), *Handbook of test development* (pp. 61–90). Mahwah, New Jersey: Lawrence Erlbaum Associates Publishers.

Reckase, M., Ju, U., & Kim, S. (2019). How adaptive is an adaptive test: Are all adaptive tests adaptive? *Journal of Computerized Adaptive Testing*, *7*(1).

Segall, D., & Davey, T. (1995). *Some new methods for content balancing adaptive tests.* (Tech. Rep.). Presented at the annual meeting of the Psychometric Society, Minneapolis.

Seo, D. G. (2017). Overview and current management of computerized adaptive testing in licensing/certification examinations. *Journal of educational evaluation for health professions*, *14*(17).

Shujaat, M. (2022, 10). Review on performance of quick sort algorithm. *International Journal of Computer Science and Information Security,*, *Vol. 19, February 2021*, 114-120.

Spray, J., & Reckase, M. (1996). Comparison of SPRT and sequential Bayes procedures for classifying examinees into two categories using a computerized test. *Journal of Educational and Behavioral Statistics*, *21*(4), 405-414.

Stocking, M. L., & Swanson, L. (1993). A method for severely constrained item selection in adaptive testing. *Applied Psychological Measurement*, *17*(3), 277-292.

Straetmans, G., & Eggen, T. (1998, 01). Computerized adaptive testing: What it is and how it works. *Educational Technology*, *38*.

Sympson, J. B., & Hetter, R. D. (1985). Controlling item-exposure rates in computerized adaptive testing. Proceedings of the 27th annual meeting of the Military Testing Association (pp. 973-977). San Diego CA: Navy Personnel Research and Development Center..

T. A. Schmitt, J. R. S., D. A. Sass, & Walker, C. M. (2010). A Monte Carlo simulation investigating the validity and reliability of ability estimation in item response theory with speeded computer adaptive tests. *International Journal of Testing*, *10*(3), 230-261.

van der Linden, W. (2022). Review of the shadow-test approach to adaptive testing. *Behaviormetrika*, *49*, 169–190.

van der Linden, W. J. (2005a). A comparison of item-selection methods for adaptive tests with content constraints. *Journal of Educational Measurement*, *42*(3).

van der Linden, W. J. (2005b). Models for adaptive test assembly. In *Linear models for optimal test design* (pp. 211–263). New York, NY: Springer New York.

van der Linden, W. J. (2005c). Solving test-assembly problems. In *Linear models for optimal test design* (pp. 77–104). New York, NY: Springer New York.

van der Linden, W. J., & Pashley, P. (2010). Item selection and ability estimation in adaptive testing. In W. J. van der Linden & C. A. Glas (Eds.), *Elements of adaptive testing* (pp. 3–30). New York, NY: Springer New York.

Veerkamp, W. J. J., & Berger, M. P. F. (1994). Some new item selection criteria for adaptive testing. *Journal of Educational Statistics*, *22*, 203 - 226.

Virginia State Statute §22.1-253.13:3. (2021).

Wainer, H., Dorans, N., Flaugher, R., Green, B., & Mislevy, R. (2000). *Computerized adaptive testing: A primer (2nd ed.).* Lawrence Erlbaum Associates Publishers.

Wang, T., & Vispoel, W. P. (1998). Properties of ability estimation methods in computerized adaptive testing. *Journal of the American Statistical Association*, *35*(2), 109-135.

Wickham, H. (2011). The split-apply-combine strategy for data analysis. *Journal of Statistical Software*, *40*(1), 1–29.

Wise, S. L. (2023). Expanding the meaning of adaptive testing to enhance validity. *Journal of Computerized Adaptive Testing*, *10*(2).

Wolter, K. M. (1985). *Introduction to variance estimation*. New York: Springer-Verlag.

Wyse, A. E., & McBride, J. R. (2021). A framework for measuring the amount of adaptation of Rasch-based computerized adaptive tests. *Journal of Educational Measurement*, *58*(1),

83-103.

Yao, L. (2019). Item selection methods for computer adaptive testing with passages. *Frontiers in psychology*, *10*(240), 109-135.

Yasuda, J., Hull, M., & Mae, N. (2022, February). Improving test security and efficiency of computerized adaptive testing for the force concept inventory. *Phys. Rev. Phys. Educ. Res.*, *18*(1).

Zheng, Y., & Chang, H.-H. (2015). On-the-fly assembled multistage adaptive testing. *Applied Psychological Measurement*, *39*(2), 104-118.