



PACIFIC METRICS  
CORPORATION

# DEVELOPING AND IMPLEMENTING A UNIVERSAL TEST ASSEMBLER

---

Wim J. van der Linden  
Michelle D. Barrett



- Different testing formats
- Assembling fixed tests
- Assembling adaptive tests
- Generalization to any test format
- Discussion



- Wouldn't it be ideal if we could assemble tests with any of the following formats, each time meeting exactly the same content specifications, for paper or online delivery, just at the push of a button?
  - Fixed linear tests
  - Linear-on-the-fly tests
  - Adaptive tests
  - Multistage tests
  - Linear-on-the-fly tests with an adaptive pretest
  - Adaptive tests with a linear pretest
  - Multistage-on-the-fly tests
  - Multistage tests with an adaptive routing test



- The methodology of automated test assembly (ATA) enables us to do so
- Historically, ATA was introduced to support various types of *fixed-form assembly* problems
- More recently, it was applied to *adaptive test assembly*
- And it is now available to assemble test with any *mixed format* in real time



- The key idea underlying ATA is its application of a methodology used in nearly every other industry, trade, or commerce to select a combination of “items” from a “pool” that
  - satisfies an (often large) set of constraints and
  - has to be optimal with respect some objective
- Prominent examples of these *combinatorial optimization problems* are: crew assignment, transportation, production scheduling, site planning, portfolio design, building an inventory, shopping, etc.



- Combinatorial optimization problems are typically solved using the technique of *mixed integer programming* (MIP) developed in Operations Research
- A typical MIP approach consists of:
  - Modeling of the optimization problem (typically using 0-1 variables)
  - Input of the model into a “MIP solver” that finds the mathematically best solution



- MIP solvers are available both as commercial and open-source software programs that
  - preprocess the problem
  - select the best algorithm to solve it
  - search for the optimal solution
  - perform various kinds of “post-optimality” analyses
- Over the last decade or so, the performance of these solvers have improved dramatically. They are now able to produce extremely fast solutions to every real-world test assembly problem



- To assist ATA users, computer interfaces have been developed which
  - automatically translate natural-language versions of any type of test specifications into the mathematical expressions required by the solver and
  - report the results back in the same natural language



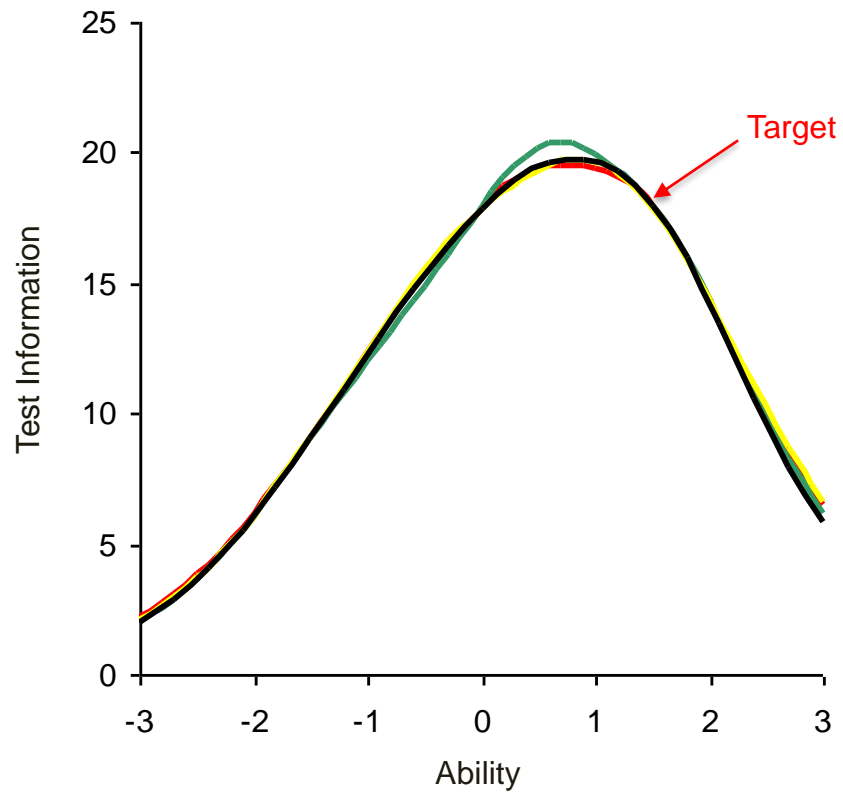


### EXAMPLES

- Assembly of three parallel test forms
  - pool of over 700 items
  - each form matching the same target information function
  - over 400 constraints to deal with all specifications



## EXAMPLES



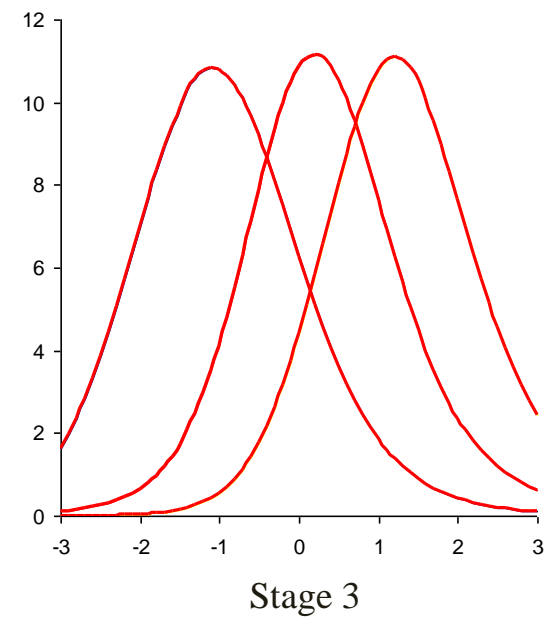
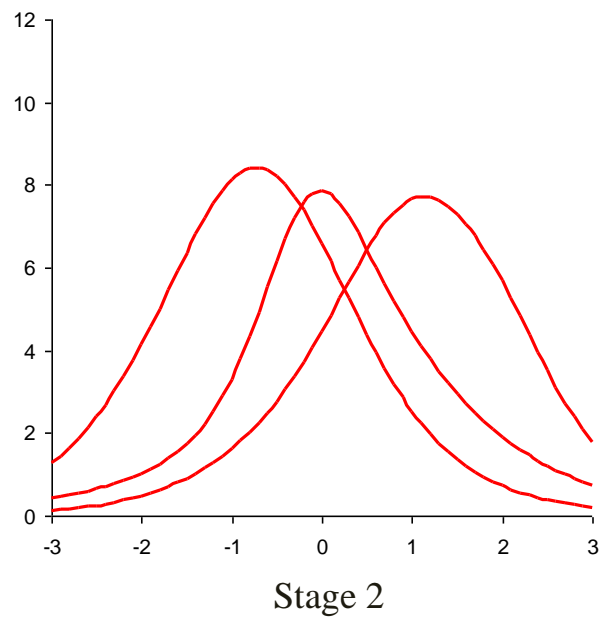
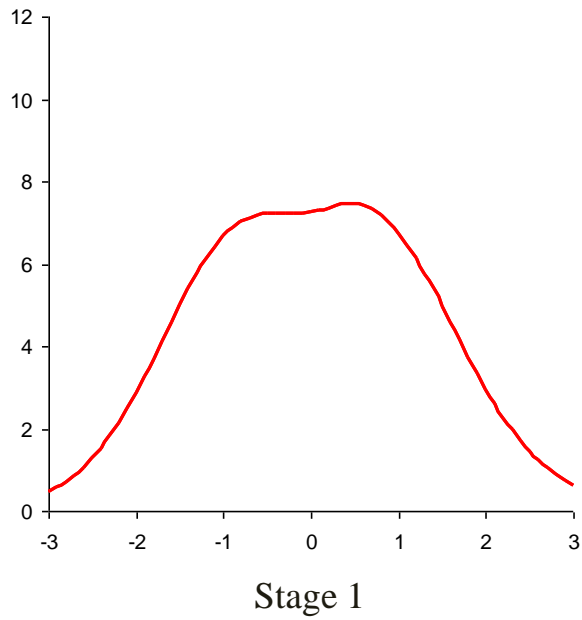


### EXAMPLES

- Assembly of subtests for a three-stage test
  - same pool and constraints
  - different target information function for each of the seven subtests
  - three runs of the solver



## EXAMPLES



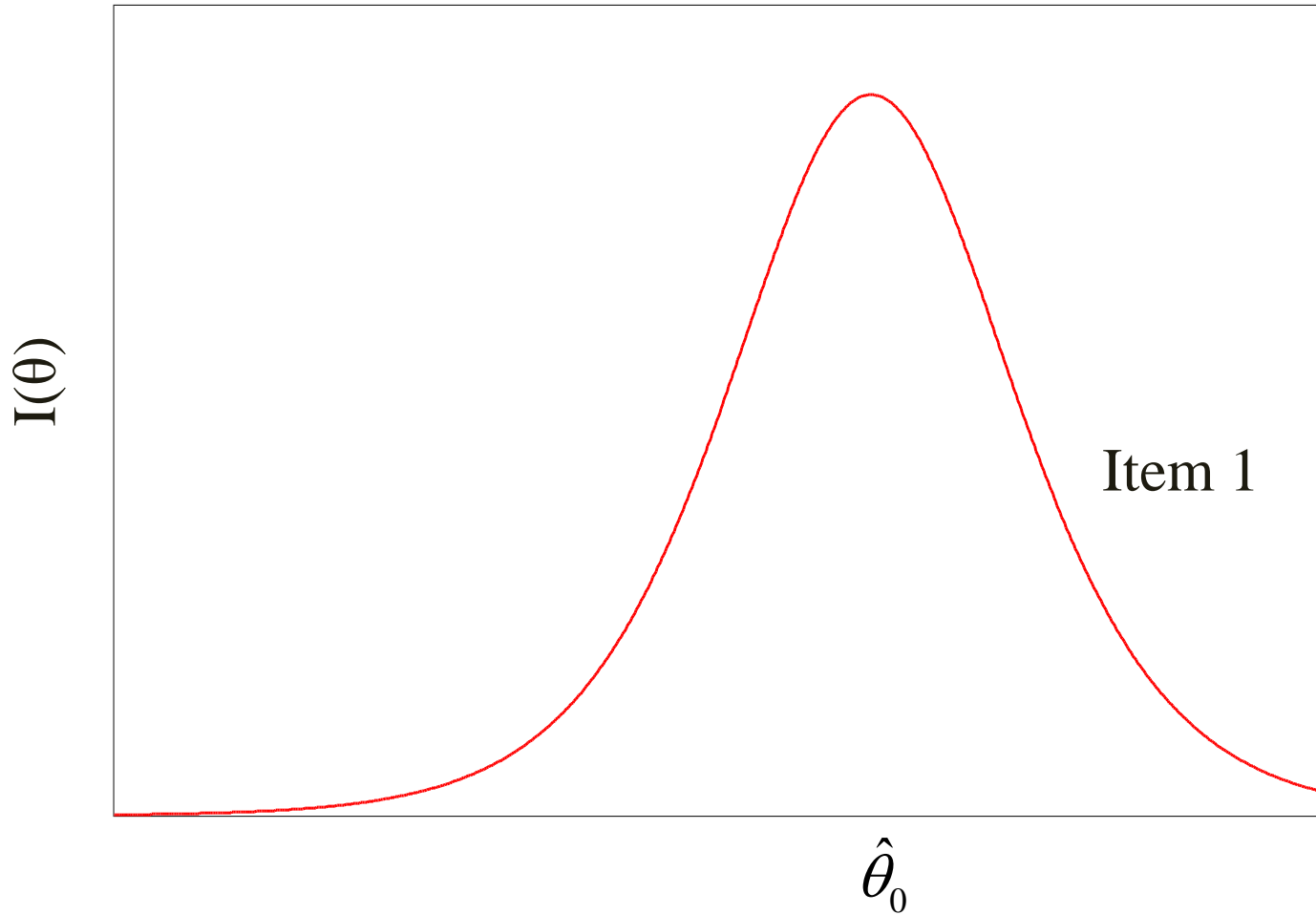


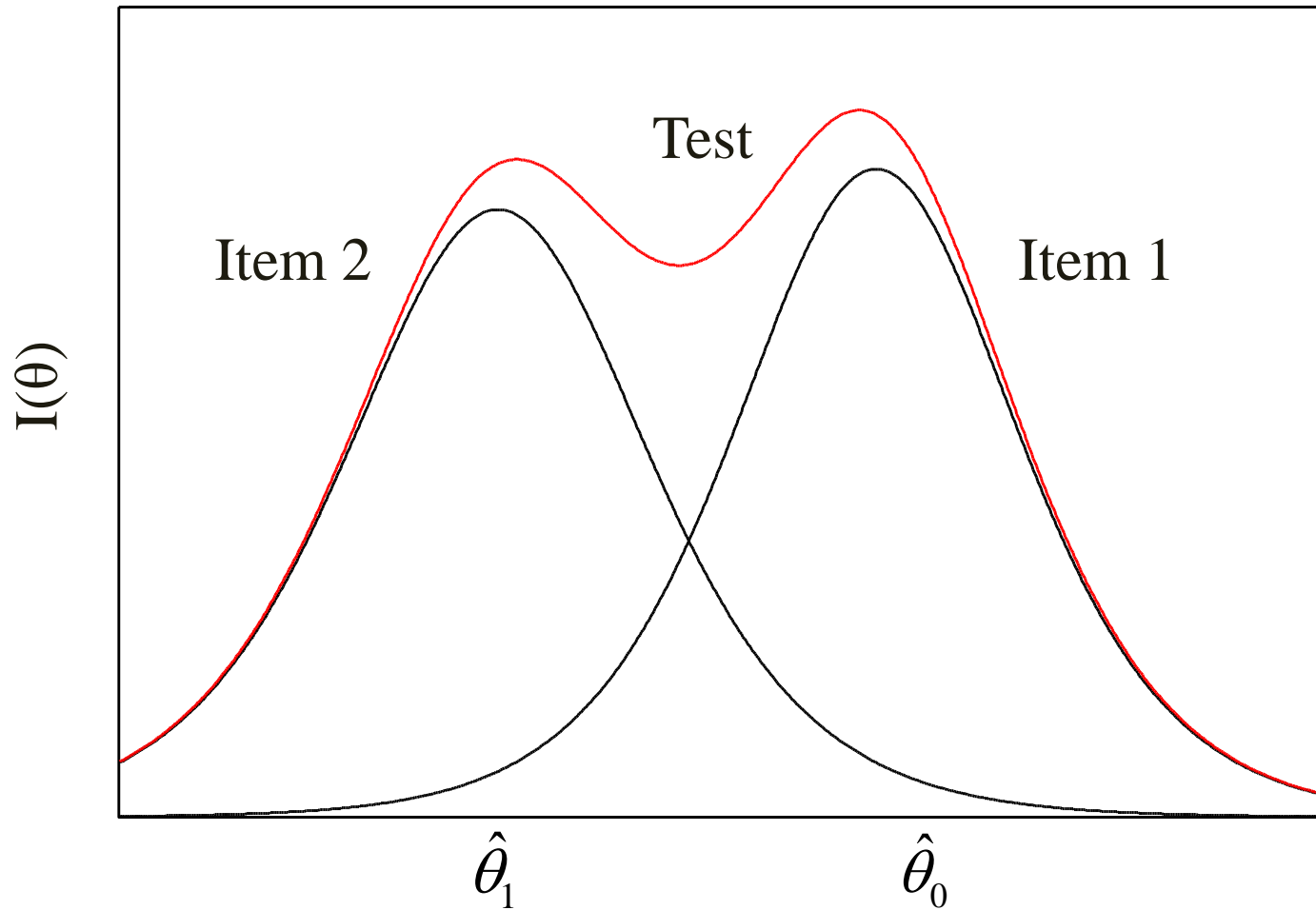
- In adaptive testing, items are selecting one at time, with each next item being maximally informative at the examinee's updated ability estimate



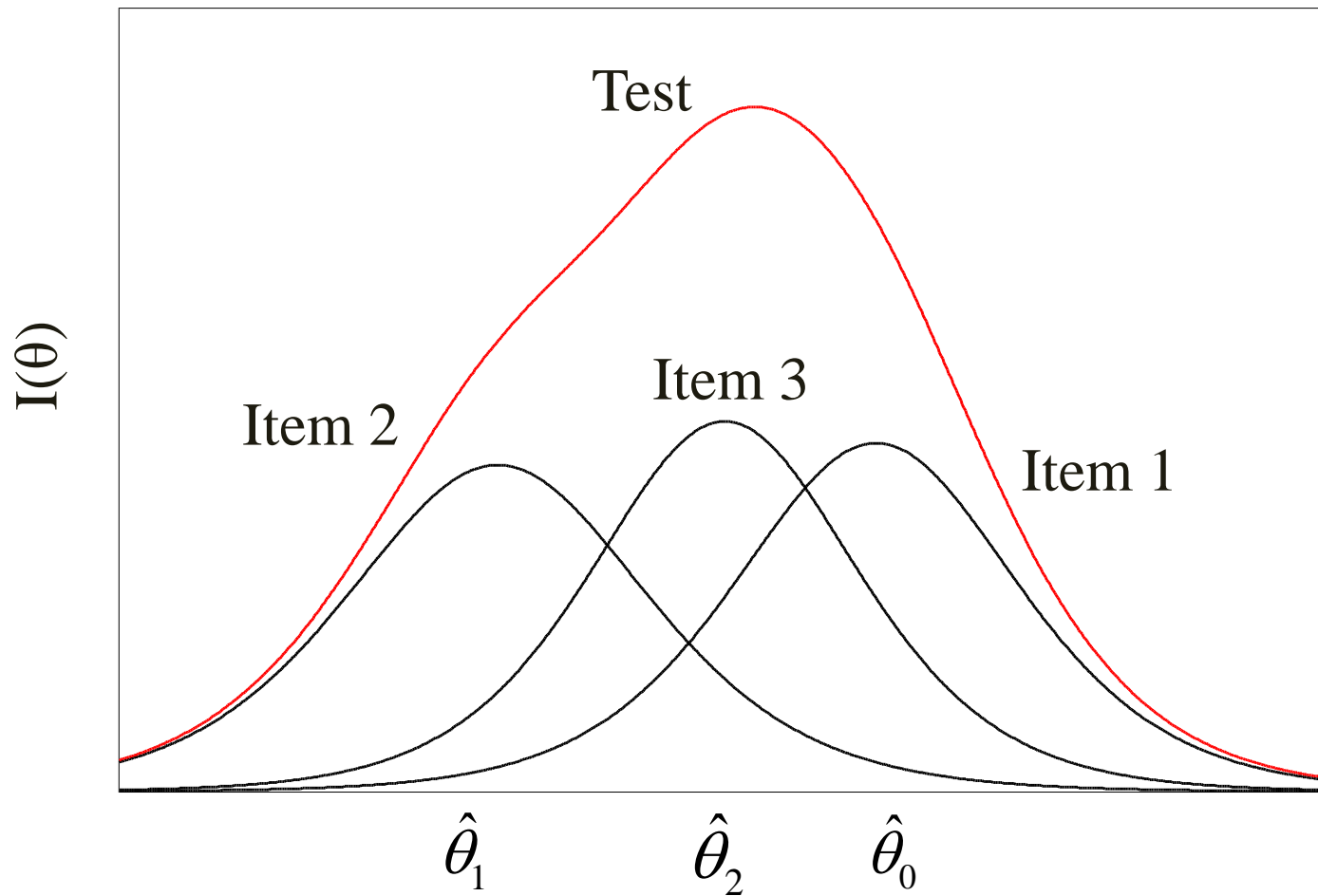
$I(\theta)$

$\hat{\theta}_0$









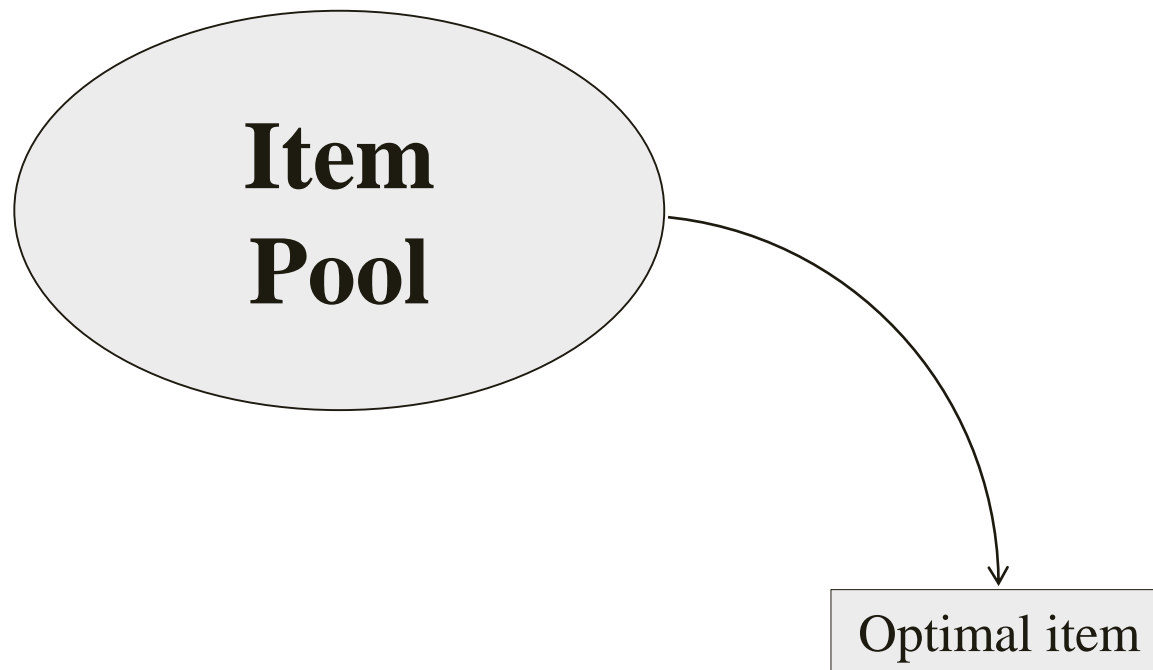


## A FUNDAMENTAL PROBLEM

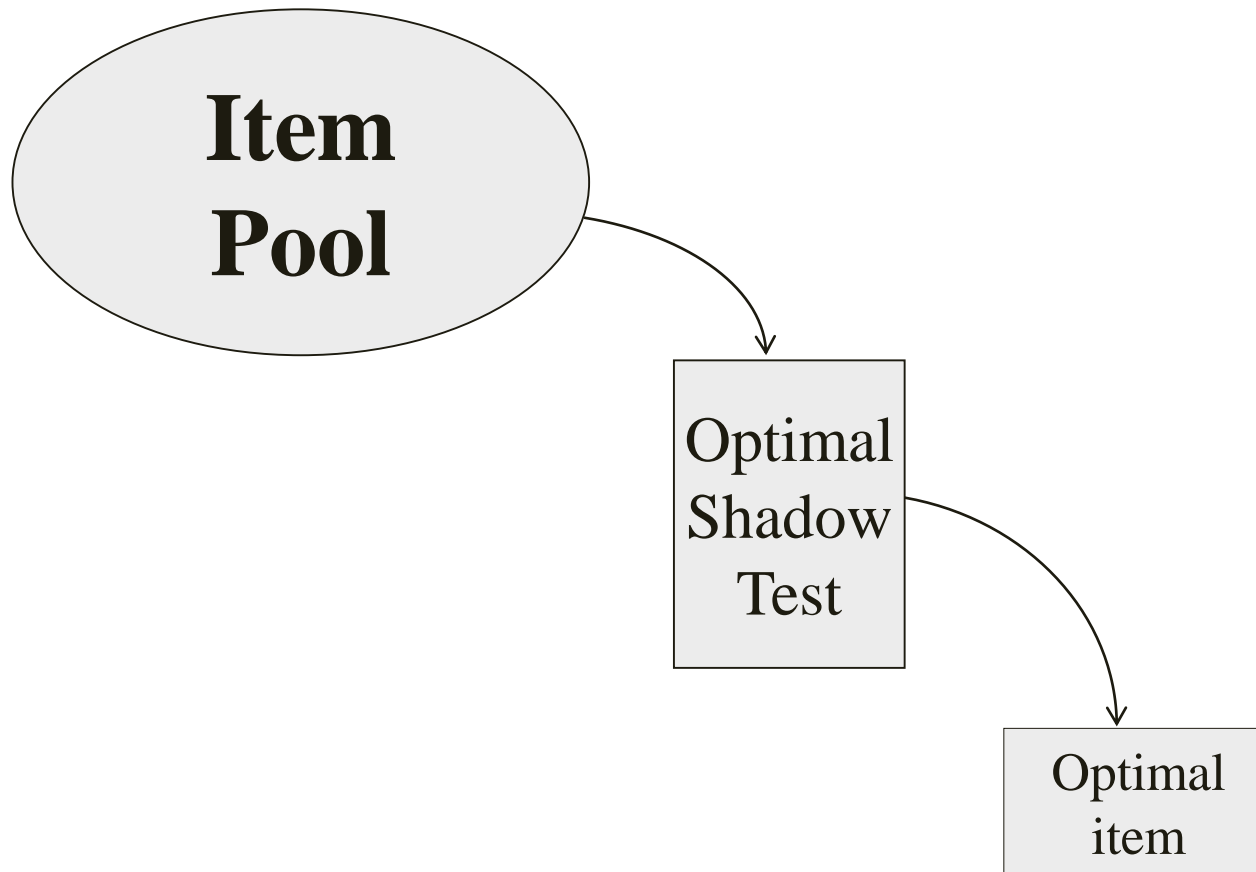
- How to satisfy all content and other constraints while selecting one item at a time?
- The solution is two-stage item selection:
  - Re-assembling of a fixed form optimal at the current ability estimate that satisfies all constraints (“shadow test”)
  - Selecting the best free item from the shadow test for administration to the examinee



## TRADITIONAL CAT

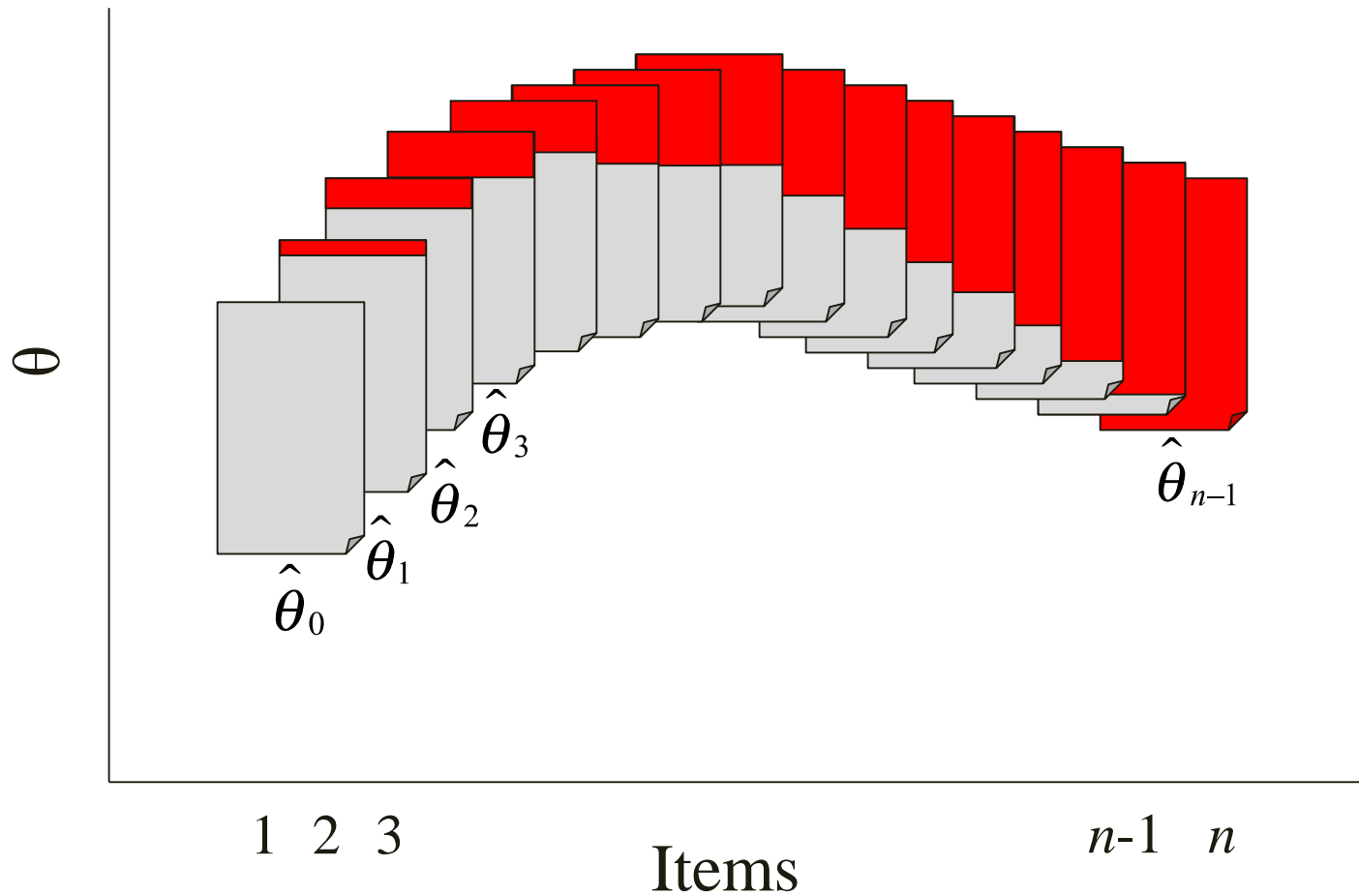


## SHADOW-TEST APPROACH

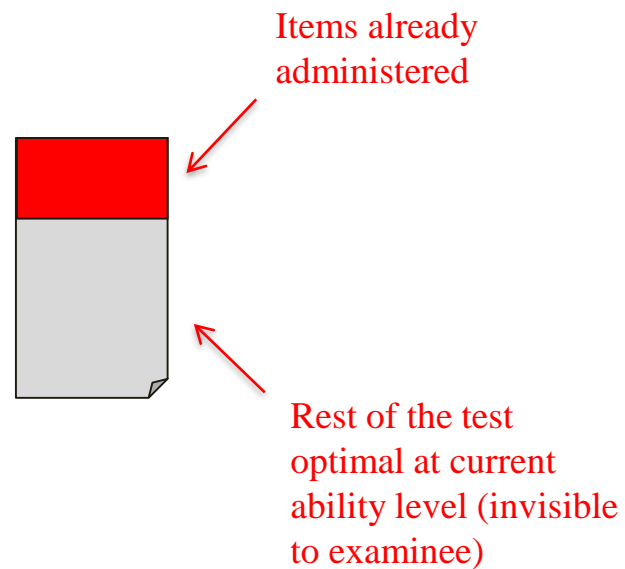




## SHADOW-TEST APPROACH

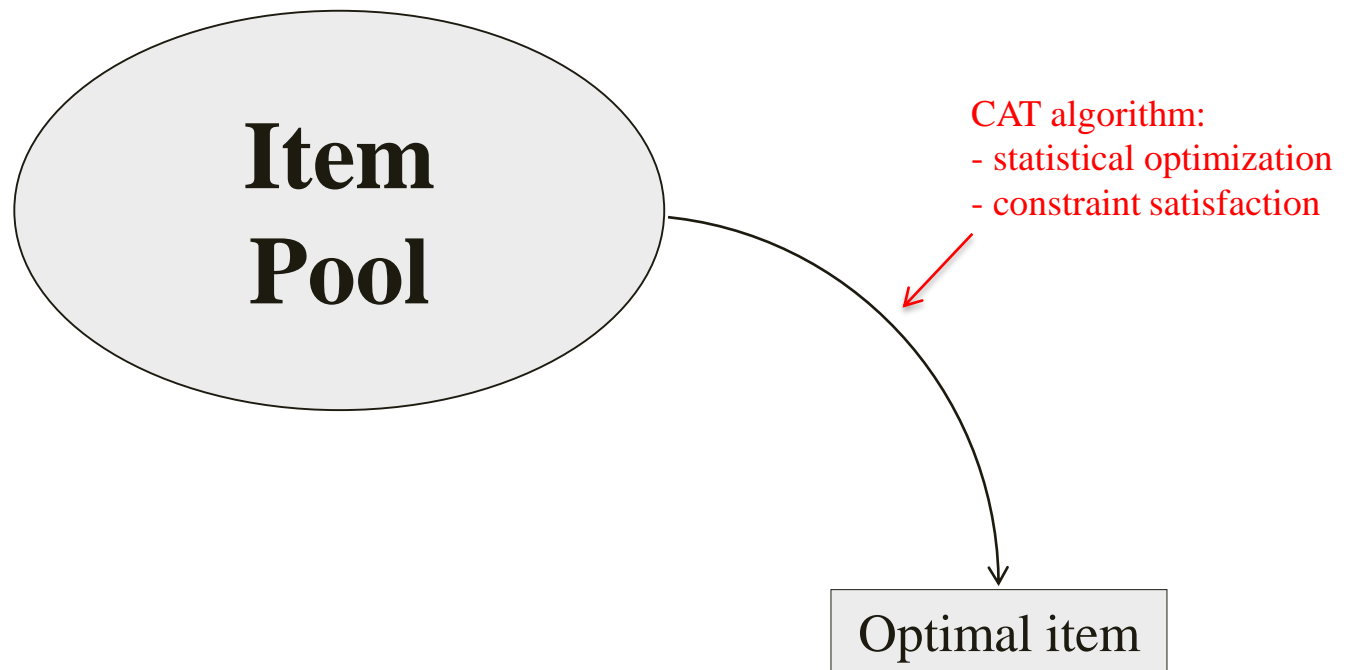


## SHADOW-TEST APPROACH



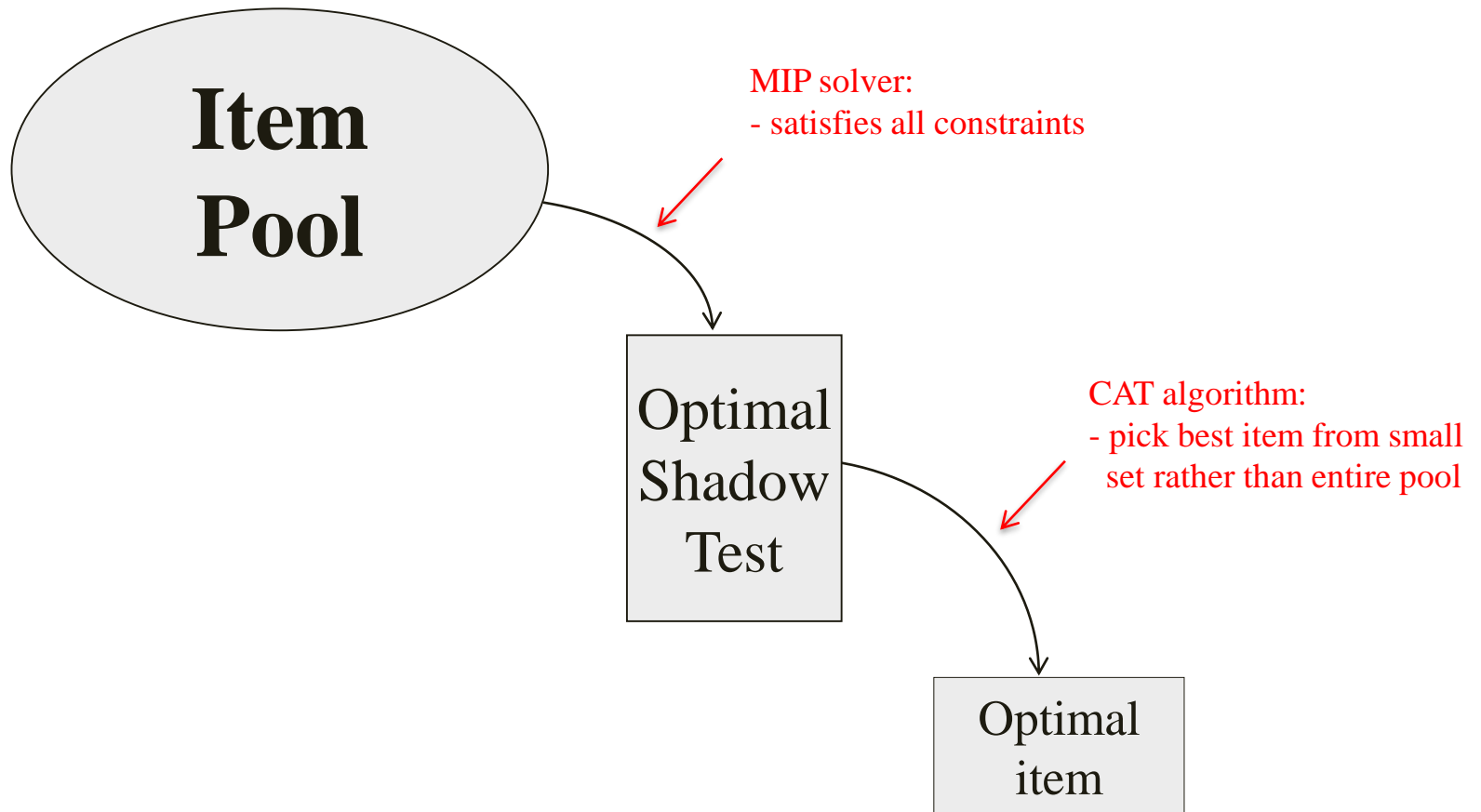


## TRADITIONAL CAT





## SHADOW-TEST APPROACH







- Once we have a shadow-test assembler, we can use it to generate any test format, just by
  - temporarily “freezing” and “thawing” the shadow test
  - reassembling the shadow test at predetermined or re-estimated ability parameters
  - using different objectives to reassemble shadow tests and to select items from it
  - Et cetera

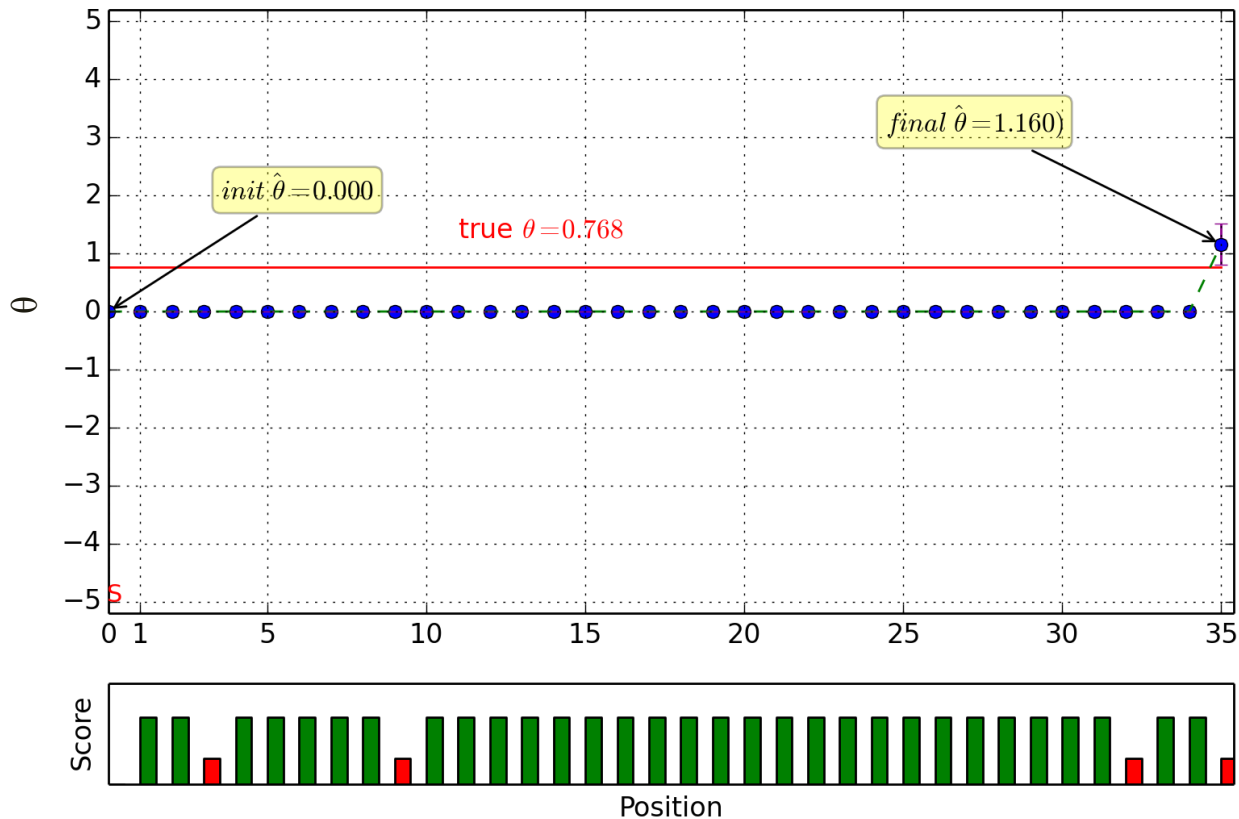


- Examples
  - *Fixed linear test*: administer first shadow test to all examinees
  - *Linear-on-the-fly test*: administer first shadow test at “estimate” of individual examinee’s ability level
  - *Multistage test*: reassemble shadow test at fixed intervals and predetermined ability levels
  - *Linear-on-the-fly with adaptive pretest*: fix shadow test at end of the pretest
  - Et cetera



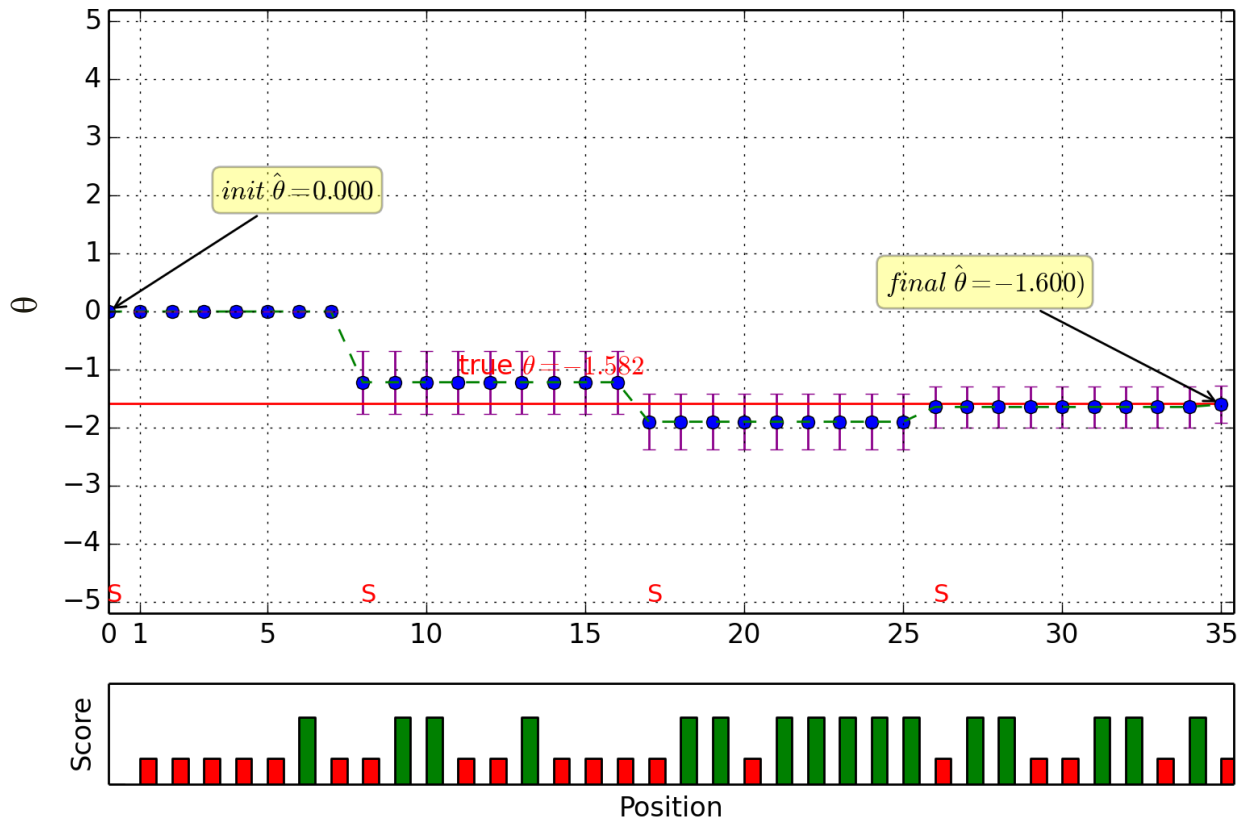
## LINEAR-ON-THE-FLY TESTING

CAT\_20140321\_031945\_586\_659 - LOFT



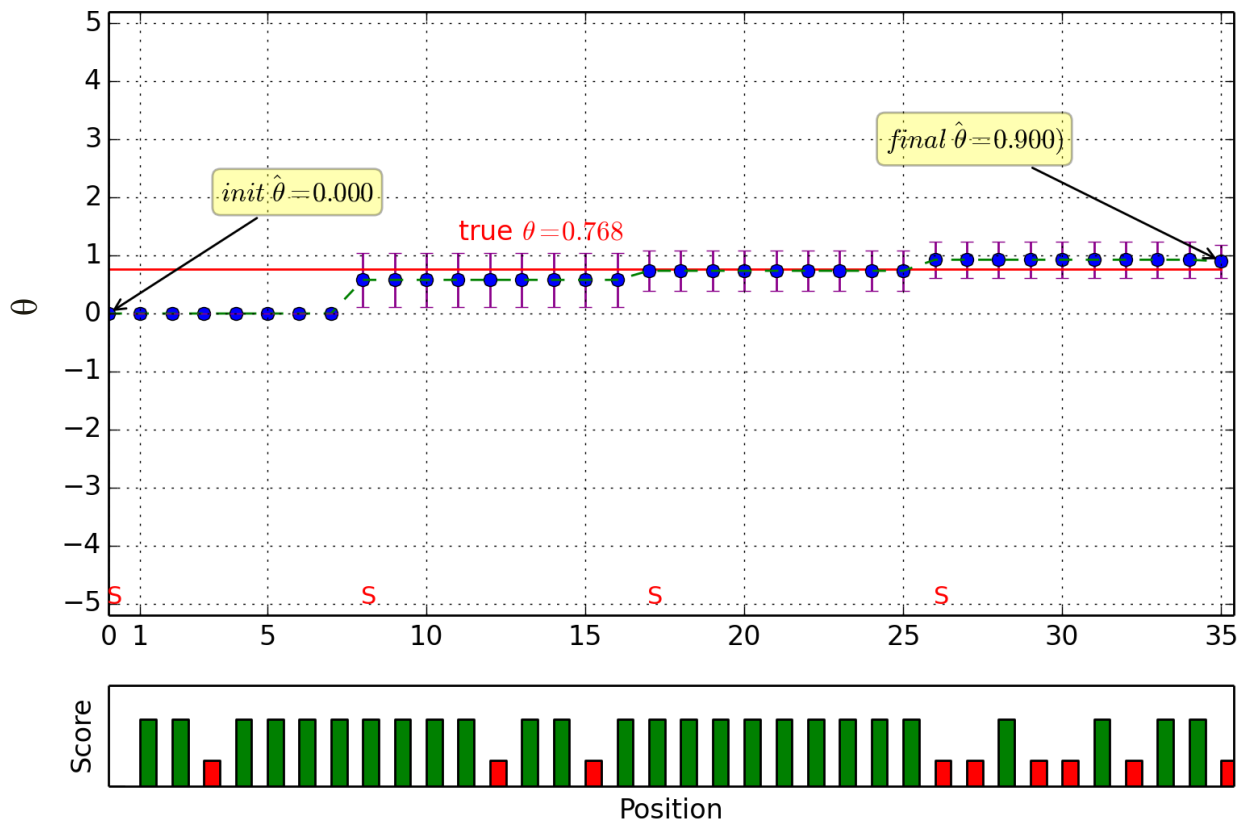
## MULTISTAGE-ON-THE-FLY TESTING

CAT\_20140321\_072208\_37\_895 - MST

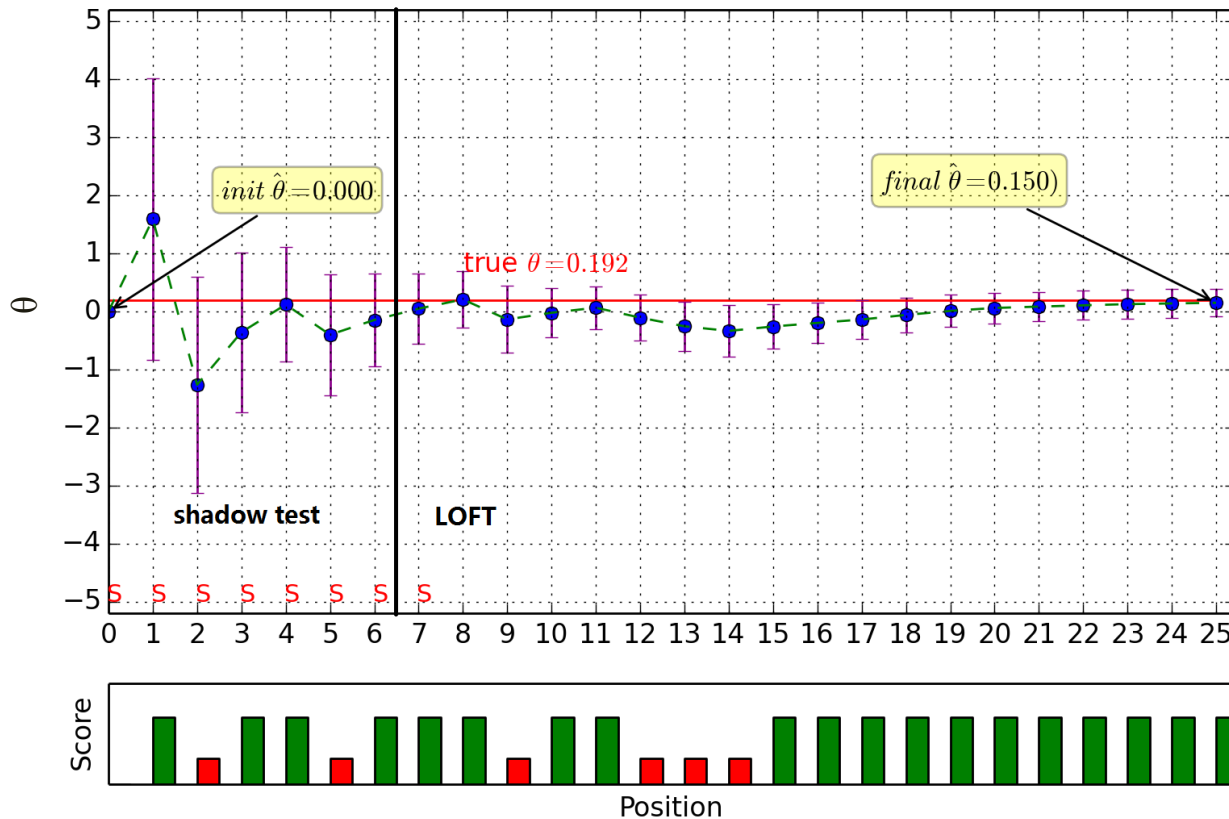


## MULTISTAGE-ON-THE-FLY TESTING

CAT\_20140321\_031933\_505\_659 - MST

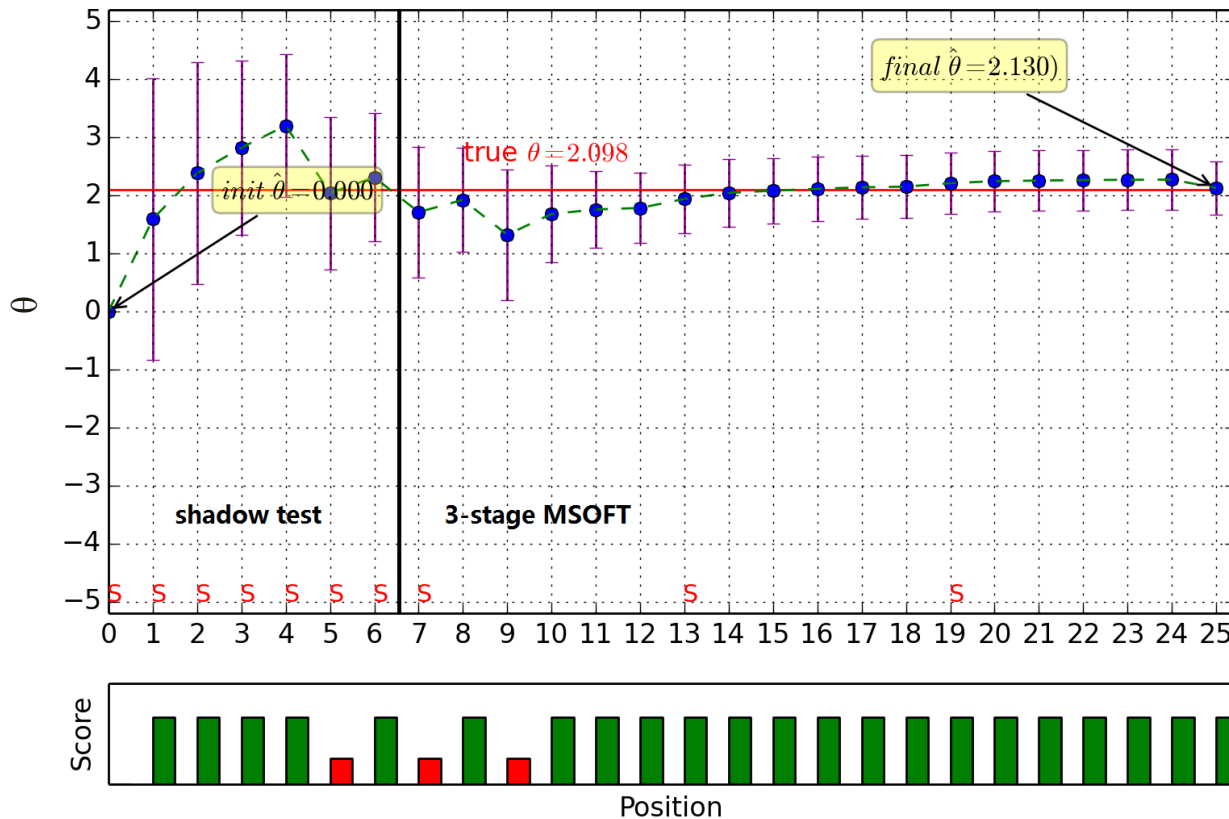


## LINEAR-ON-THE-FLY WITH ADAPTIVE ROUTING TEST





## MULTISTAGE-ON-THE-FLY WITH ADAPTIVE ROUTING TEST



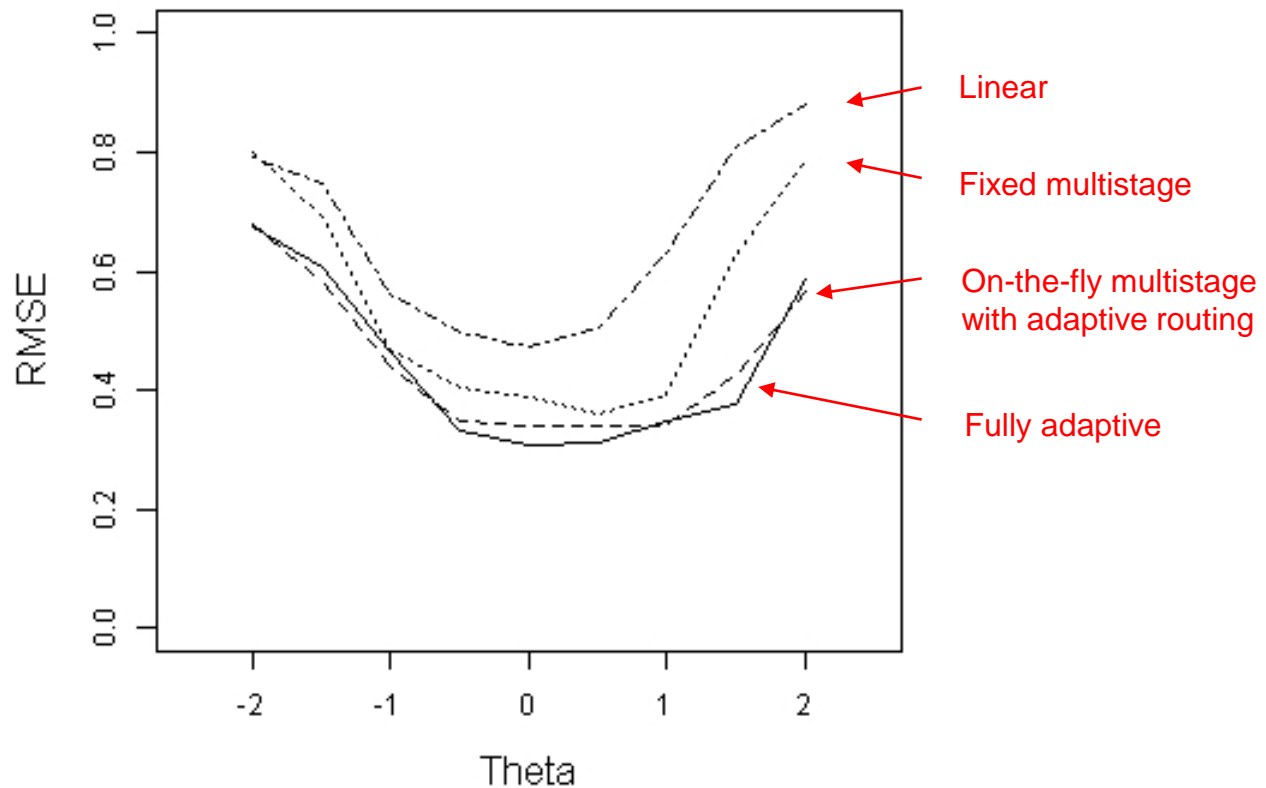


- Each format can be assembled to satisfy exactly the *same* set of content specifications.
- So, the same testing program can be offered in multiple modes without violating any of the content specifications for any of the examinees
- Generally, more accurate scores are obtained for formats with
  - longer adaptive elements
  - that occur earlier in the test





## COMPARISON OF FORMATS





A Universal Test Assembler should be

- Useable
- Scalable
- Available
- Extensible
- Interoperable.



- Useable for people
  - Assessment editors or developers
  - Psychometricians
  - Anyone who needs to validate the assembly





- And also useable for machines
  - Content authoring platforms
  - Test delivery platforms



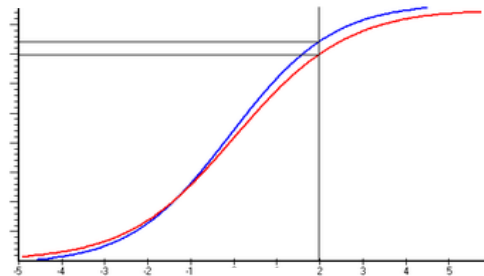


- Across multiple test assembly problems that may differ in:
  - Data needed for assembly
    - Item and stimulus metadata
    - Item statistics

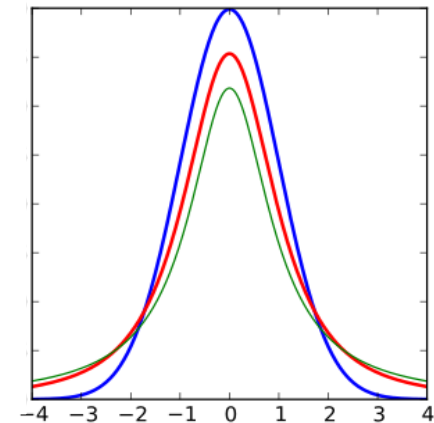
Param	C-Param	A-Param-S	B-Param-S	C-Param-S	Precludes	P-value	Ptbis	Content 1	Content 2	Content 3	Content 4	Word Cou	Answer Ke	Depth of k	DIF Flag	Requires c	In
1.1679	0.228141	0.086388	0.267482	0.076823		0.301037	0.347587	Math	Lev1	Functions	Build	36	D	3	NONE	NO	NO
1.94464	0.150238	0.041495	0.196851	0.070164		0.436901	0.459045	Math	Lev3	Geometry	Rt Triangle	15	A	1	NONE	NO	YES
342234	0.268595	0.100722	0.112991	0.035221		0.73243	0.351952	Math	Lev1	Functions	Exponenti	25	C	1	NONE	NO	NO
1.10864	0.218247	0.097616	0.06335	0.050158		0.573577	0.379333	Math	Lev2	Statistics	Data	35	D	3	NONE	NO	YES
1.51813	0.172618	0.071791	0.190477	0.059172		0.407791	0.365908	Math	Lev2	Algebra	Equations	20	D	1	NONE	NO	YES
1.27368	0.201745	0.044841	0.423458	0.094583		0.357462	0.312033	Math	Lev3	Functions	Interpret	43	D	3	NONE	NO	YES
1.07171	0.222779	0.090515	0.332568	0.068225		0.466862	0.322962	Math	Lev1	Algebra	Expression	44	D	2	NONE	NO	YES
162912	0.369897	0.124406	0.137627	0.025022		0.594613	0.335059	Math	Lev1	Algebra	Polynomial	25	B	3	NONE	NO	YES
1.55989	0.229764	0.052579	0.155978	0.046322		0.439445	0.433163	Math	Lev1	Algebra	Equations	38	D	1	NONE	YES	YES
1085733	0.135862	0.107071	0.100622	0.03096		0.584981	0.394228	Math	Lev1	Functions	Linear	36	C	1	NONE	YES	YES
1.18606	0.310699	0.072017	0.354083	0.062697		0.483012	0.390495	Math	Lev1	Algebra	Inequalities	23	C	2	NONE	NO	YES
1.26077	0.167647	0.060419	0.151671	0.070361		0.347721	0.379705	Math	Lev1	Functions	Interpret	32	B	2	NONE	YES	YES
1.34153	0.165005	0.047878	0.216526	0.068061		0.468445	0.323384	Math	Lev1	NumbQua	Vectors ar	28	C	1	NONE	NO	YES
1.84408	0.261442	0.07182	0.175335	0.069216		0.426773	0.431011	Math	Lev1	Functions	Build	18	B	2	NONE	NO	NO
108637	0.297983	0.077894	0.109959	0.018892		0.626353	0.362423	Math	Lev1	NumbQua	Vectors ar	35	B	2	NONE	YES	YES
1.15901	0.170498	0.094843	0.15471	0.058359		0.394419	0.381174	Math	Lev3	Functions	Interpret	37	C	1	NONE	NO	YES
1.27216	0.228095	0.057655	0.188062	0.067612		0.450552	0.308504	Math	Lev1	Functions	Interpret	40	C	1	NONE	YES	YES



- Across multiple test assembly problems that may differ in:
  - Objective function
    - Maximum information
    - Match to existing test information or test characteristic curve
    - Optimal field test allocation
    - ...



© Pacific Metrics 2010



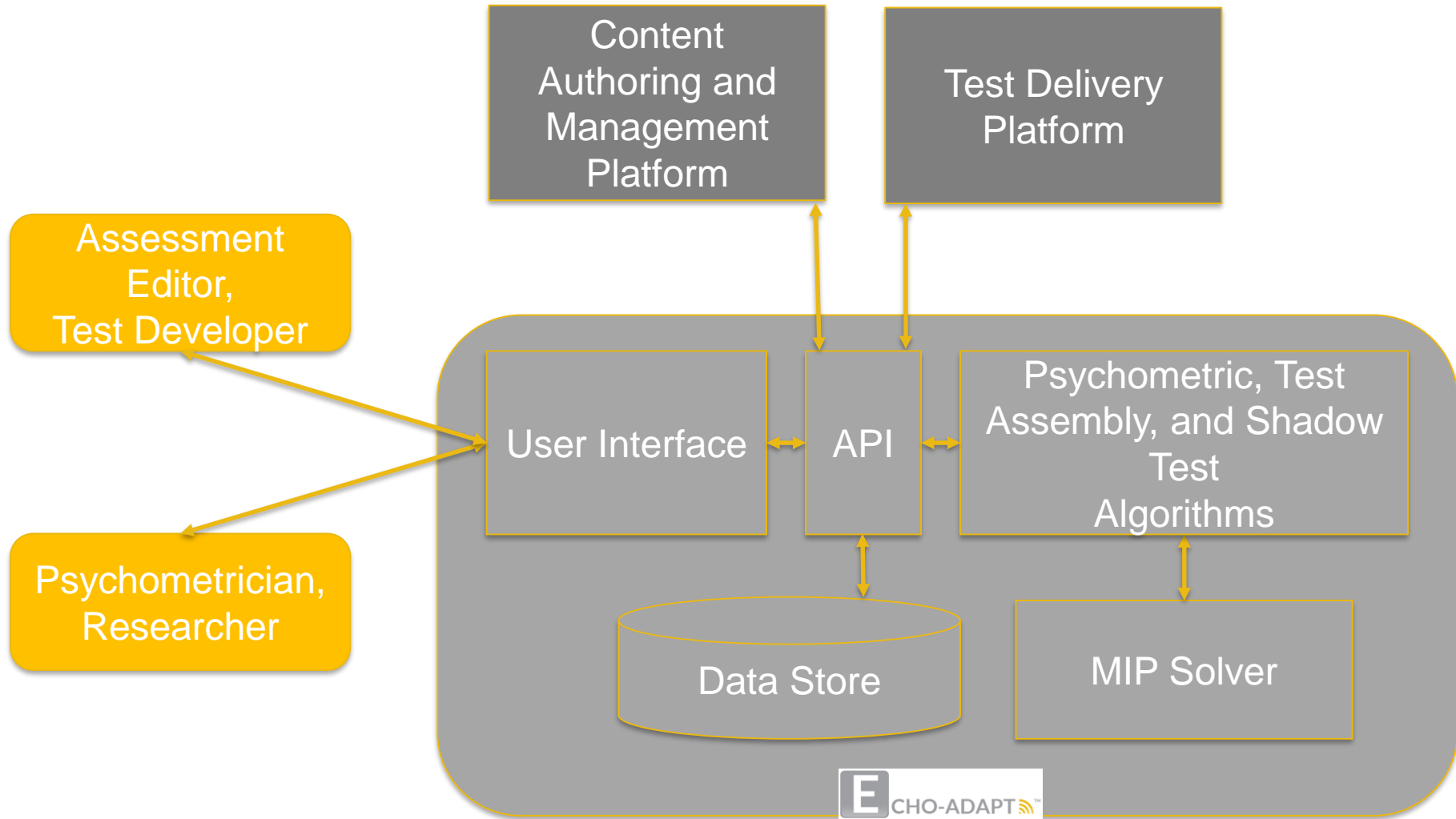


- Across multiple test assembly problems that may differ in:
  - Desired output
    - Real-time assembly (adaptive)
    - Package for test form review
    - Single versus multiple test forms



```
NextItemsAndSessionState:
  description: This is what CAT engine produces. next_items
  required:
    - nextItems
    - numberOfItemsInNextStage
  properties:
    nextItems:
      $ref: "#/definitions/IdList"
    numberOfItemsInNextStage:
      type: integer
    linear:
      type: boolean
      default: True
    assesmentResult:
      $ref: "#/definitions/assesmentResult"
    sessionState:
      $ref: "#/definitions/SessionState"

NewSessionIdNextItemsAndSessionState:
  required:
    - testTakerSessionId
  allOf:
```







- User Experience / User Interface Challenges
  - Use item pool with any metadata fields
  - Flexible constraint specification
    - Must be able to identify characteristics of items to constrain
    - Known: Required queries are difficult for users to specify
  - Feedback to user
    - Over-constrained situations
    - Level of adaptability given constraints

- User Experience / User Interface
  - Enable immediate research and validation with user-driven simulation capability

### Simulation Settings

**Number of Test Takers**   
Enter between 1 and 10000 Test Takers

**True Ability Distribution** ⓘ


**Distribution** ☒ Normal ☐ Uniform

Mean	Std. Deviation	Lower Bound	Upper Bound
<input type="text" value="0"/>	<input type="text" value="1"/>	<input type="text" value="-4"/>	<input type="text" value="4"/>

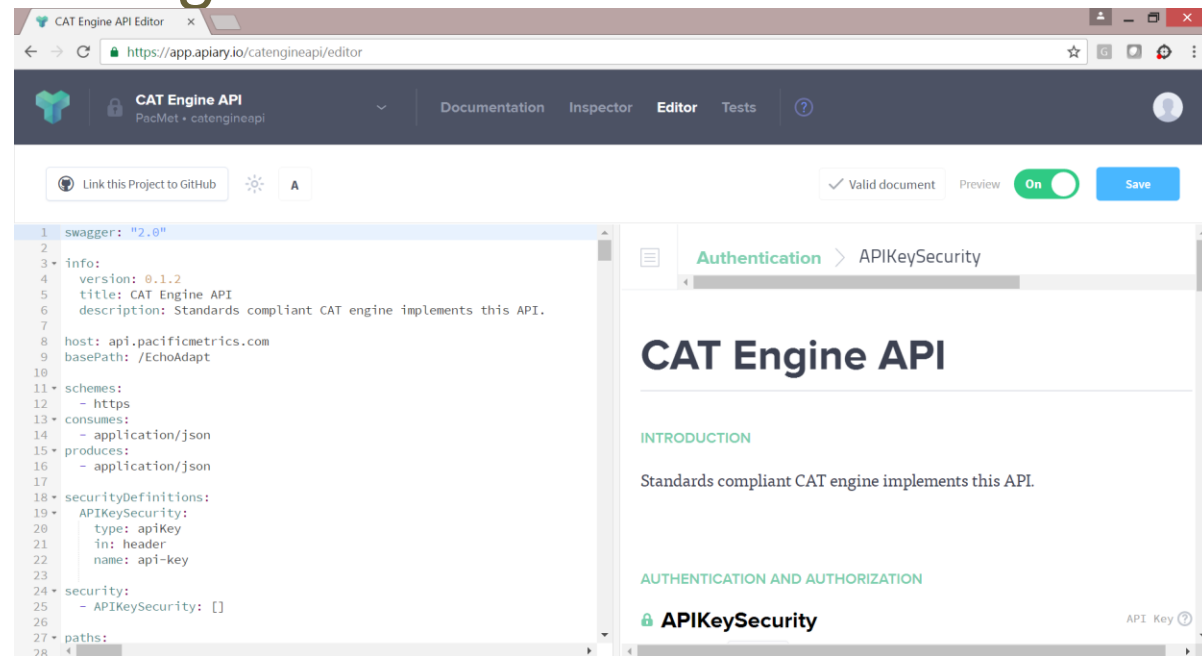
[Reset](#)

### Results

Success! Simulation ran on MM/DD/YYYY, HH:MM:SS

 **Download Results as CSV**

- Integration with test delivery and content authoring
  - Published APIs
  - Stubs for testing





- Scalable

- Human

- Level of specialization required to configure and select an optimal test



- Machine

- Concurrent test assemblies
    - Concurrent test-takers
    - Low latency





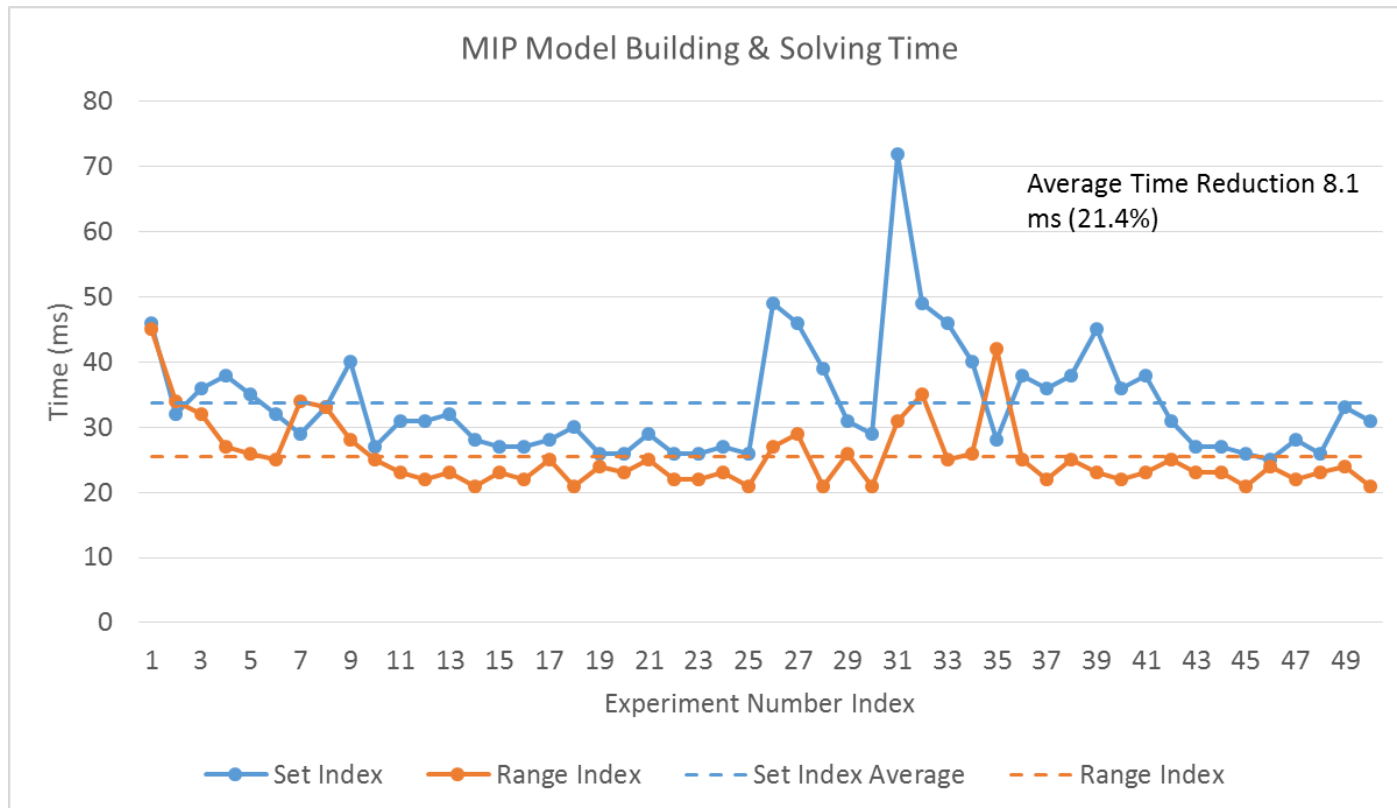
- Human
  - Employ operations research scientist to build the software, not to construct individual models
  - Provide the right level of abstraction for flexible model building
  - Design the user interface so mathematical models can be built based on simple and intuitive user input



- Machine
  - Mixed Integer Programming Solver
    - Open source versus commercial
      - GPLK, LP Solve
      - IBM CPLEX, Gurobi, FICO Xpress
    - License for cloud deployment
    - Implementation considerations for low latency
    - Implementation considerations for large problems

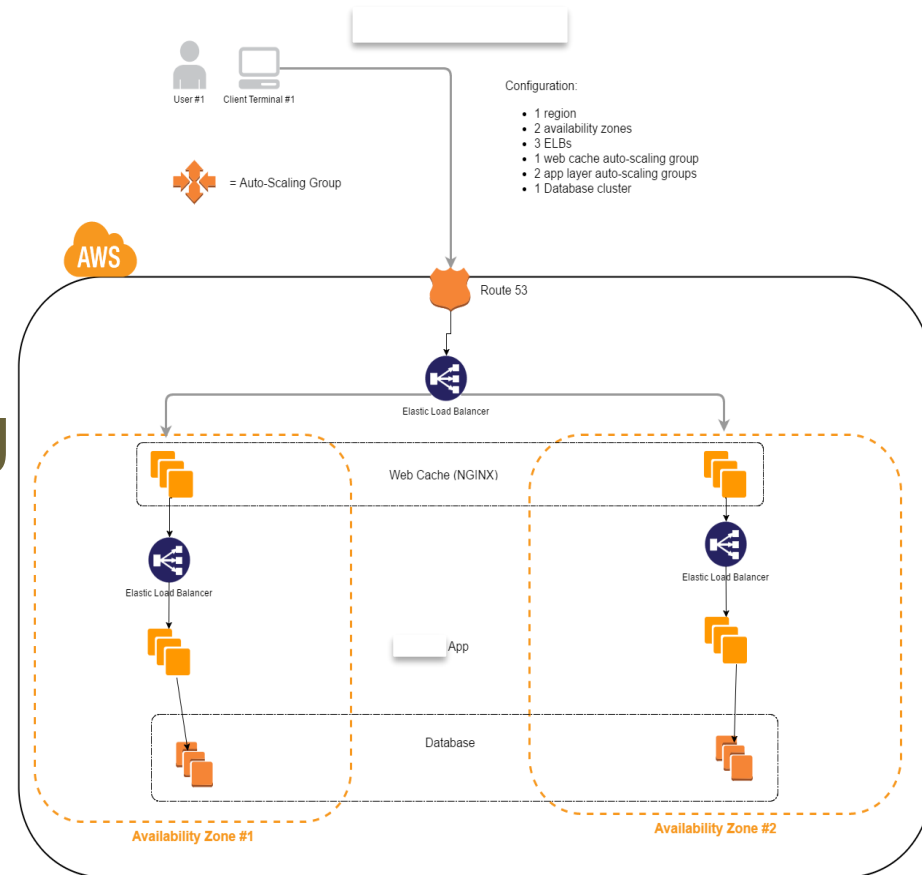


- Machine
- Latency





- Machine
  - Distributed
    - Caching
    - Load balancing
    - Parallel computing
    - Data store



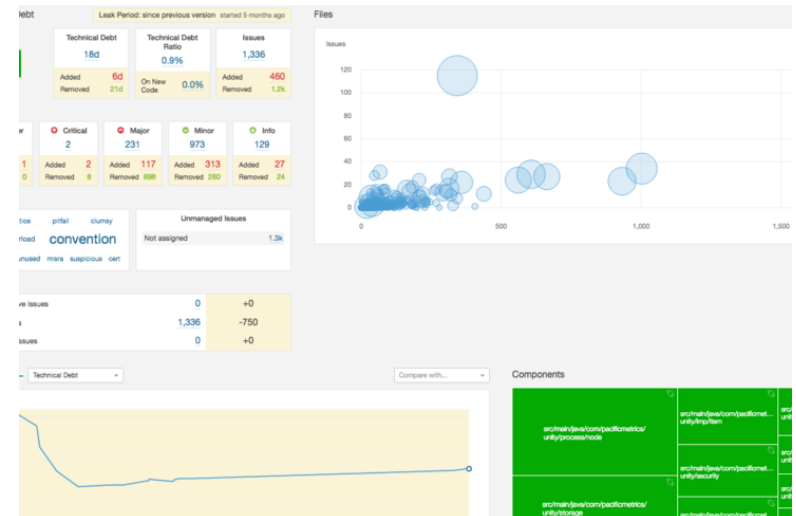




- Available
  - Up when authoring platforms need it
  - Up when test delivery platforms need it
  - Cloud deployed
  - Fault-tolerant
    - Multiple availability zones
    - Multiple regions
    - For adaptive testing, return of complete shadow test at each adaptive step



- Extensible
  - How quickly new features can be implemented
  - Monitored quality of code
    - Unit test coverage, automated unit tests, integration tests, regression tests
  - Flexible data structures





- Interoperable
  - Industry-standard data exchange to interact with item authoring and test delivery platforms
    - IMS Global working on standard for adaptive testing engines (as subset of QTI)
    - IMS QTI assessment packages
  - QTI metadata and QTI usage statistics



Current Registrations:  
[imscert.org](http://imscert.org)



Join us on our journey as we implement a universal test assembler that is:

- Useable
- Scalable
- Available
- Extensible
- Interoperable.

Echo-Adapt™ beta release: December 2016



mbarrett@pacificmetrics.com  
wvanderlinden@pacificmetrics.com  
hgarner@pacificmetrics.com