A Process for Blaise Data Emulation of Complex Instruments

Todd Flannery and Ed Dolbow, Westat, USA

Limitations of traditional testing

- As instruments grow in length and complexity, testing is complicated by the relationship among thousands of data items and the routes that they generate. This limits the coverage of scenario and specification testing
- Specification testing has to account for thousands of combinations of responses.
- Scenario testing is limited by the difficulty of developing data covering every combination even with the assistance of a planned scenario as a starting point.
- Although regression testing can generate data for routes executed in prior rounds, any number of routes could be unaccounted for because they were never used before.

Test Methods

- Data emulation can assist in producing a test base that is more comprehensive and more representative of all possible combinations of responses and routes found in the survey instrument.
- Combinatorial logic software can produce a comprehensive input file.
- We implemented an approach using these tools on the Population Assessment of Tobacco and Health (PATH) adult survey, a large national longitudinal study with annual data collection cycles.
- The adult instrument contains 1500 items and approximately 2000 or more response variables.

Preparing for data emulation

- The route to a key question early in the survey depends on 17 responses or non-responses to earlier questions in the same survey or previous rounds to fully test the route to the question. There are quite a few questions of equal complexity in the instrument.
- Using the Blaise emulator tool, we executed this instrument thousands of times and produced large datasets of randomly generated values that abided by the Blaise rules.
- The Blaise software emulates item responses but not the inputs to the instrument. The inputs or preloads have dependencies or constraints that are not enforced in the Blaise rules.

Developing pre-loads, i.e. input variables

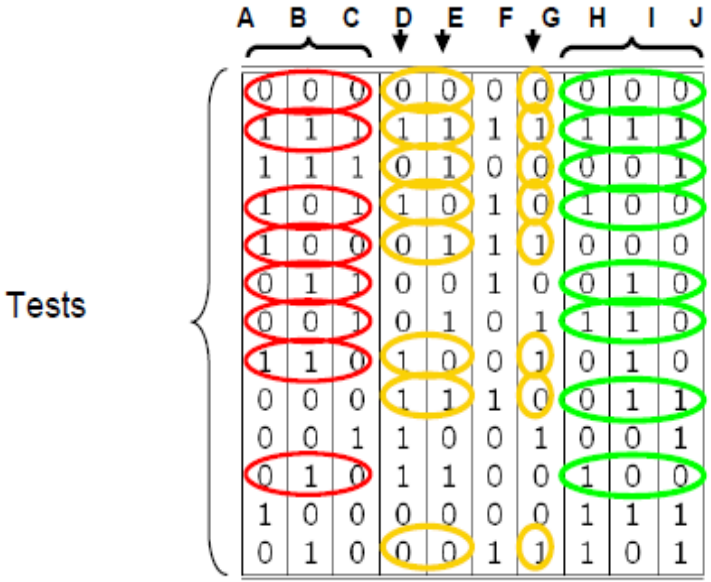
- The PATH instrument has many pre-loads (inputs) derived from previous rounds of data collection or from other instruments administered during the same round.
- We utilized two approaches to generate these preloads.
  - > Create preload records/cases using combinatorial methods.
  - > Use longitudinal preloads from the previous round of data collection.

Method 1 – generating preload data with ACTS

- Automated Combinatorial Testing for Software (ACTS)
  - > Developed by National Institute of Standards and Technology (NIST).
  - > Addresses limitations on creating tests that provide complete coverage due to the interactions between two or more variables.
  - > Generates multi-factor combinations of data items to determine interactions expecting some of them to cause unique failures.
  - > A more straightforward example than PATH follows.

Method 1 ACTS

In this example, 10 variables have two values. There are 1024 unique combinations of these values. The ACTS software uses trios of data items (the columns) to cover several combinations on the same row and requires only 13 records to cover all of the three way interactions. Each row can represent a row of input prior to emulating record-level random data.



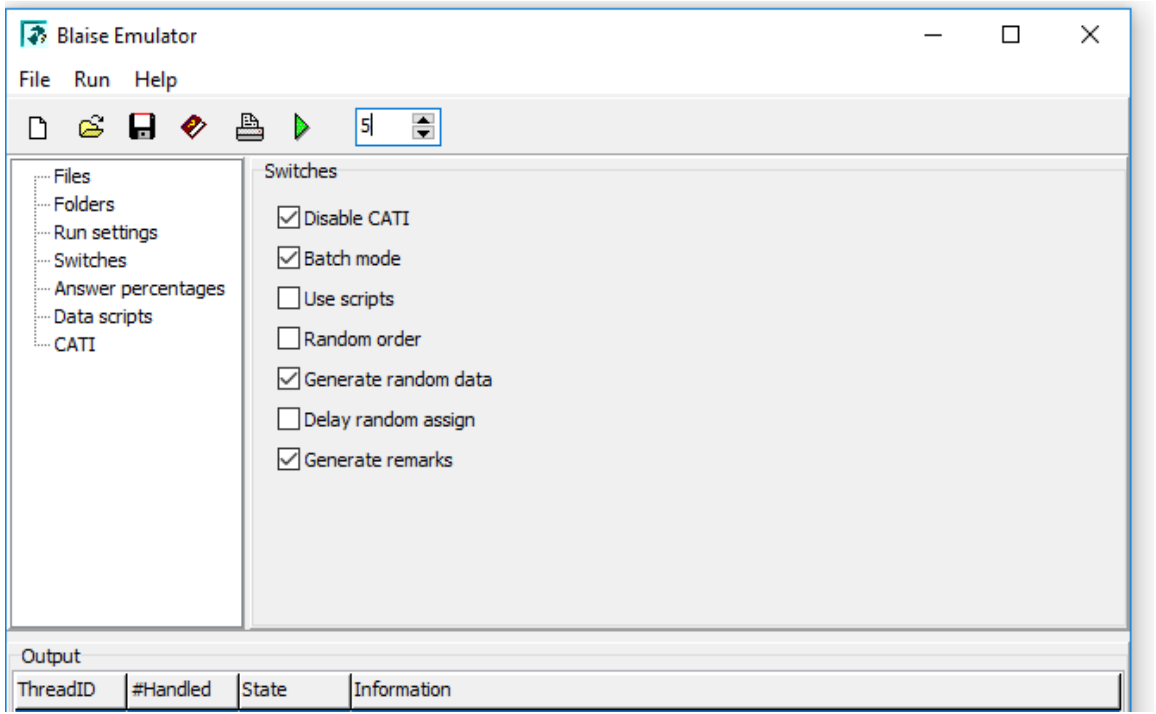
D. Richard Kuhn, Raghu N. Kacker, Yu Lei. "Practical Combinatorial Testing" NIST Special Publications 800-142, October, 2010

Method 2 – pre-existing preload data

- Longitudinal studies can use existing input data from an earlier wave to mimic combinations of input data that are likely to occur in the planned data collection.
- It takes a several months to develop these variables for each wave and the schedule insists that we testing before the files become available.
- Instruments use responses generated in other questionnaires administered during the same visit. Pre-loads from previous rounds cannot account for this condition.
- Limited in that we may miss unusual routes that did not occur in the first part of the wave.

The Blaise Emulator

- BTEMULA.EXE and BLEMU.EXE (multi-threaded)
- Generates random values for all missing FIELDs on the route.
- Applies values from formatted text scripts to the Blaise data.



What could possibly go wrong?

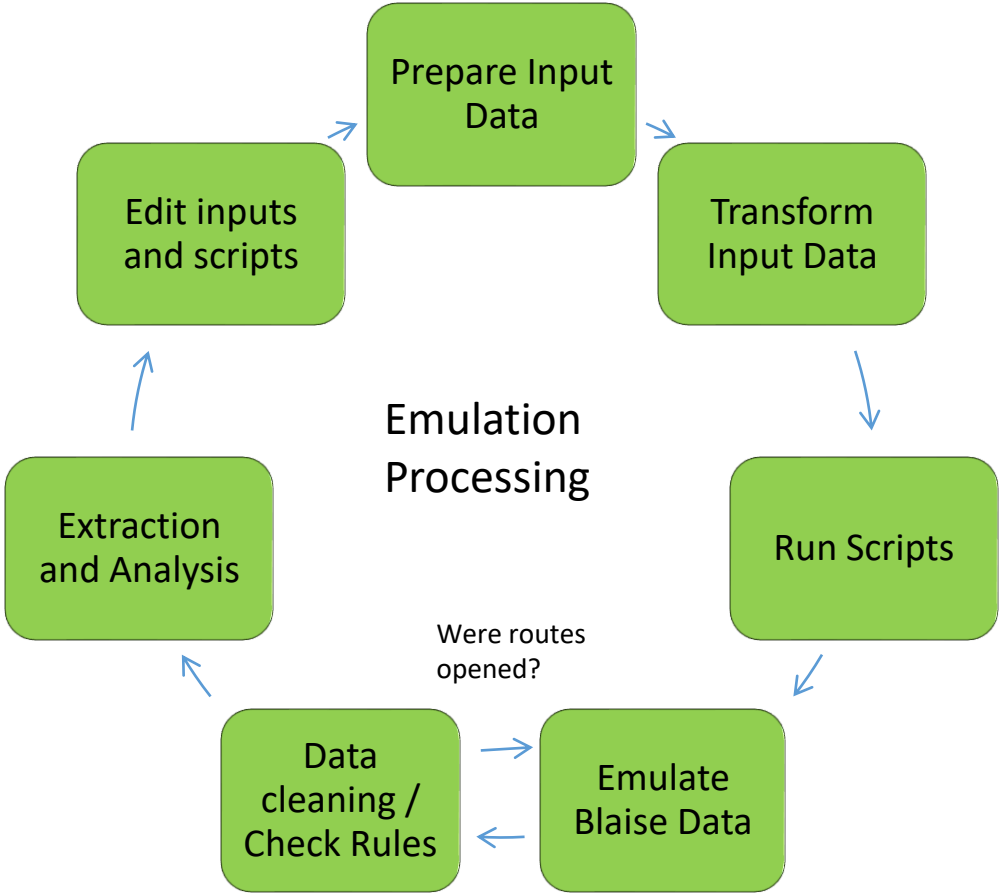
- If rules do not enforce data integrity, the emulator will not either.
- Randomizing some items, such as (dates, times, or text strings) usually produces meaningless data. But, routes can be impacted.
- Eligibility based on screening or sampling algorithms may be impacted, since all values are equally likely.
- Values are generated based on the definition of the field, not hard or soft edits.
- External coding via Maniplus or using blank strings to hide responses does not exclude values from being selected.

Mitigating errors

- We apply tools to mitigate each of these issues and produce large-scale data that can provide valuable information regarding the integrity of our code or specifications.
- Tools are applied as steps in an iterative process.

The Emulator Process

- Iterative, in that each previous cycle informs the next cycle.
- Variation of data and routes is desired.
- Limitations and constraints are applied judiciously based on the project needs.



Results

- First pass through process:
  - > Inputs were data entered by a tester via DEP, copied several times over.
  - > Most value obtained in the refinement of the scripts and edits.
  - > 15,000 test records, each containing 4000 columns of Blaise data, 3 working days to process.
- Second pass:
  - > Data imported from earlier wave.
  - > Analysis provided hundreds of results for further investigation.
  - > 12,500 records analyzed.
- Third pass:
  - > ACTS tool generated inputs using 4-factor combinations.
  - > Process refinement allows greater input from experts.
  - > Emulated about 6,000 records for analysis.

Conclusions

- Emulation provides greater coverage of routes in testing.
- Process refinement will produce some valuable results.
  - > Elimination of some of the tedious scenario-testing tasks
  - > Analysis of test data can begin in parallel with a testing cycle
- There are several areas where we can improve the process
- Scalable and adaptable for Blaise 5

Thank You

For more information on NIST combinatorial testing:

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.227.9998&rep=rep1&type=pdf

https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-142.pdf

For more information on ACTS software:

https://csrc.nist.gov/projects/automated-combinatorial-testing-for-software/downloadable-tools#acts