Towards Automatic Experimentation of Educational Knowledge

Yun-En Liu[1], **Travis Mandel**[1], **Emma Brunskill**[2], **and Zoran Popović**[1]
[1]Center for Game Science, Computer Science & Engineering, University of Washington
[2]School of Computer Science, Carnegie Mellon University
{yunliu, tmandel, zoran}@cs.washington.edu, ebrun@cs.cmu.edu

ABSTRACT
We present a general automatic experimentation and hypothesis generation framework that utilizes a large set of users to explore the effects of different parts of an intervention parameter space on any objective function. We also incorporate importance sampling, allowing us to run these automatic experiments even if we cannot give out the exact intervention distributions that we want. To show the utility of this framework, we present an implementation in the domain of fractions and numberlines, using an online educational game as the source of players. Our system is able to automatically explore the parameter space and generate hypotheses about what types of numberlines lead to maximal short-term transfer; testing on a separate dataset shows the most promising hypotheses are valid. We briefly discuss our results in the context of the wider educational literature, showing that one of our results is not explained by current research on multiple fraction representations, thus proving our ability to generate potentially interesting hypotheses to test.

Author Keywords
Datamining; Games; Education

ACM Classification Keywords
H.5.0 Information interfaces and presentation: General

INTRODUCTION
Many disciplines have experienced an explosion of data in the past decade, transforming the way we do science [8]. Web-based software has led to a similar increase in data for the behavioral sciences. This is particularly exciting in these domains, as subjects are often costly, difficult to recruit, and may not be demographically diverse [15]. In the past, lack of subjects has often meant only sparse coverage of experimental spaces due. But now, for particular branches of the behavioral sciences in which humans can both remotely perform interesting tasks and are willing to do so, the increase in data means that we now have the potential to learn much more about how humans interact with software. But to take full advantage of these users, we first need effective tools to allow us to explore these new huge datasets.

To this end, we present an automatic experimentation and hypothesis generation framework designed for these big data scenarios. In the basic framework, the inputs are a target objective function on users, such as learning gains, and a set of factors that form a parameter space of possible experimental conditions, such as different learning interventions. We automatically bin users into experimental conditions, identifying the parameters with broadest impacts averaged across other factors. It then recurses on the best parameter setting as measured by the objective function, and finds the best setting of the remaining parameters, providing confidence intervals at each stage. This means both that we automate much of the experimental process, and also provide a much more thorough coverage of the hypothesis space. This frees the researcher to perform tasks that humans do best: deep data analysis and generation of hypothesis spaces for the system to explore.

Unfortunately, it is often the case that we do not possess full control over the user experience. For example, software companies may not want to expose many users to highly risky experimental conditions. Or, in a more extreme case, we may want to analyze already-collected data in a purely offline manner. To deal with these situations, our full framework uses importance sampling to simulate the desired user distributions given data drawn from a different distribution. This allows us to ask many different questions on already-collected data, allowing us to fully utilize previously collected data.

We demonstrate the power of our proposed framework in the educational domain, by implementing and running it offline on an data set with a sampling distribution different than the desired one. We show how our method can generate hypotheses about which parameters and their settings are best for encouraging near-transfer, and confirm these hypotheses with statistical tests on a completely different dataset. Some of our results match current educational theories, but some do not, suggesting further experiments to run either online or in a classroom. Of course, our framework has important limitations: for example, our greedy search may not explore effective parts of the hypothesis space in the presence of parameter interactions, or the data may be so noisy that we require prohibitive numbers of players to discover interesting information. Still, our ability to automatically find interesting parts of the hypotheses space suggests that this method may become a useful tool in behavioral research.

CHI 2014, April 26–May 1, 2014, Toronto, Ontario, Canada.

http://dx.doi.org/10.1145/2556288.2557392

RELATED WORK

Online experimentation

Major companies such as Microsoft [19] and Amazon [20] have performed online experiments for years. For researchers, Mechanical Turk has proven to be quite helpful to many scientists looking for cheap, high-quality user data [18]. Games have also become an increasingly popular source for behavioral data, and have been used to study the effects of optional rewards [2] and tutorials [4]. More fine-grained methods, such as multi-armed bandits, have also been used in online settings to maximize click through rates of search results or article recommendations [25].

These web-based mass experimentation platforms have a few key benefits. Mechanical Turk has been shown to provide inexpensive, reliable results and has a demographic spread much wider than typical pools for social science research [10]. In the games domain, the observed behavior is "in the wild" [3], increasing external validity [33]. We build on this work by proposing a framework that automatically runs series of experiments depending on intermediate results, and use importance sampling to estimate the results when the sampling distribution does not match the desired one.

Scientific Discovery

Researchers in AI have been working for years to develop systems capable of generating scientific knowledge. This field, known as scientific discovery, has generated many such systems aimed at automating different scientific behaviors [21]. For example, Lee et al. used a feedback loop between the RL rule induction program and expert knowledge to identify potentially carcinogenic compounts [23]. Perhaps the most comprehensive example of such a system is Robot Scientist Adam, a fully automated robot capable of the full loop of hypothesis generation, experimental design, and data analysis in yeast genomics [17].

We focus on the automatic selection of experiments and introduce a new algorithm for choosing which ones to run. We differ in our source of data: automatically running experiments on humans introduces many problems not present in a laboratory setting. In some settings, such as education, there may be many experimental variables: instruction duration, number representation, ordering of concepts, problem type, hinting systems, etc.. In addition, we often want to both find general rules about how different factors affect student learning, but also the specific settings that optimize rate of learning and maximum transfer ability. We deal with both objectives with a greedy search strategy designed to find the "good" parts of the hypothesis space.

Educational Data Mining

Our example application runs experiments in the educational domain to identify factors contributing to variation in learning, a common theme in the educational data mining community. Intelligent tutoring systems, especially cognitive tutors [5], have been used for many experiments: for example, Rau et al. [29] test the usefulness of multiple fraction representations with self-explanation compared to single representations. Or in the educational games domain, Lomas et al. [26] used an educational game to run two large-scale experiments with many conditions on the effect of challenge on motivation and learning. We are not proposing a new tutor or game, but rather a method that gathers or uses existing data from some source (such as a tutor or game) to automatically run experiments with many factors. To the best of our knowledge, other EDM researchers have not proposed or used an automatic method for choosing and running experiments in a hypothesis space; nor have they used importance sampling to run multiple experiments on the same dataset with a different sampling distribution than desired.

BASIC FRAMEWORK

Here we present a simplified version of our framework, which requires full control over how players are sampled and is designed to run one experiment at a time. We will be investigating how varying the presentation of (fraction) number lines affects users' ability to answer future number lines. Fraction number line problems typically take the form of a line representing the reals, with at least two points marked for scale, and ask where new fractions should be placed or what fraction corresponds to some point. In the next section, we will extend this framework with importance sampling to allow us to run multiple analyses on the same dataset, and then present an implemented version of it that was able to automatically run parameter searches to find which types of number lines lead to maximal near-transfer to new number line questions, using data from an online educational game.

First, let's consider how a researcher might run an experiment.

1. The researcher hypothesizes about how one or more factors might impact a variable of interest, ex. different fraction representations or hinting systems might lead to different performance on future number lines.
2. If subjects are expensive, she cannot test more than a few of these factors simultaneously. Instead, she must decide which factors are most interesting: perhaps she compares a few different representations with no hinting systems.
3. She decides on an experimental procedure to test the role of this factor on the variable of interest, including whatever assessments are necessary. Here, this will likely involve the choice and refinement of existing number line tests, whether or not the players should participate online or come into the lab, and so on.
4. She runs the experiment, then collects and analyzes data.
5. Assuming the results become well-established and accepted, eventually another researcher may pick other factors to investigate, holding the already-studied ones constant. For example, perhaps symbolic representations are better in the initial experiment. Then the next experiment might test the effects of different hinting systems for symbolic fraction number lines.

From a research perspective, perhaps the most interesting part of this process is data analysis and hypothesis generation. The rest is needed to both gather data confirming or rejecting

the hypothesis, and to ensure that there is sufficient statistical power even with a small number of subjects.

However, assume now that we have a constant stream of new users: say, several thousand per day. Furthermore, assume that they are interacting with a system under the experimenter's full control and are willing to participate in the experimental conditions and take any assessments necessary. Then, there is no difficulty in finding subjects, and the experimenter does not have to be present to run experiments. Furthermore, with so many users and control over the assignment of players to conditions, we can test many experimental conditions, not just a handful. Thus, the experimenter need not carefully select factor levels: she can simply specify the available factors and the system can explore them to identify the ones that seem most informative. Finally, statistical analysis becomes easier with clean experimental designs, though as we will show later we can continue to operate even if we do not have full control over which interventions are chosen.

Of course, the total number of experiments possible is combinatorial in the number of factors, so it is necessary to choose a search strategy. In this paper, we propose a greedy search: at each step, we choose a parameter and its setting that leads to the best performance when randomly selecting other parameters, then recurse on the remaining parameters. This leads to selecting parameters which are broadly effective at the top of the experiment tree (due to the randomization of other parameters), but which quickly narrows (due to the greedy setting of parameters with each step). This strategy is appropriate in a domain such as education where we both want to create generalizable knowledge about which dimensions of the parameter space are most effective, and also optimize some metric like learning gain. In other domains, such as psychology, the goal may simply be to find the factors that cause the largest differences between settings and so a different search strategy may be needed.

We formalize our framework in the following way. The experimenter specifies a *parameter space* P she is interested in exploring. The parameters that make up this space correspond to factors, in standard statistical parlance. Restricting ourselves to categorical variables for the time being, each parameter $p_i \in P$ has k_{p_i} settings $s_{p_i j}$, $j = 1 \ldots k_{p_i}$, which correspond to factor levels. In addition, she specifies an *experimental set* E, $E \subset P$: these are parameters she wants the system to investigate. Letting our population of players be represented as X, she also provides an objective function $f : X \to R$ that produces a real value for each user.

Our algorithm can be thought of as a greedy depth-first search over the experimental set E, marginalizing over the parameters $P - E$. We describe it as follows.

- First, we test each setting of each parameter. To test a setting means to randomly assign some number of players N to an experimental condition with $p_i = s_{ij}$ and with all remaining variables $P - p_i$ set randomly.

- Each experimental condition C_q is specified exactly by a parameter $p_q \in E$ and its setting $s_{p_q j_q}$. Since there is one condition for each parameter setting, we have $q = 1 \ldots \sum_{p \in E} k_p$. Let X_q be the set of players assigned to C_q. Let $F_q = \frac{1}{N} \sum_{x \in X_q} f(x)$.

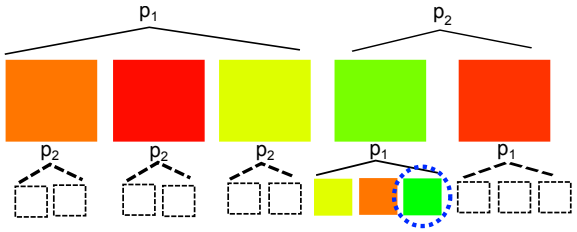


Figure 1. An illustration of the tree built by our algorithm with two parameters, p_1 and p_2. We first explore all individual parameter settings at the top of the tree, holding one parameter fixed to some value and randomizing over the other parameter. We order the nodes by function value and recurse, so that we hold p_2 to its best setting and vary p_1. In this work we stop once all nodes have been set for the first time, though with an exponential number of users we could explore all nodes.

- Once each condition has N players, we order the C_q from greatest to least by their associated objective values, F_q. We store the best node seen so far as B. We recurse on each condition in order, with parameter space $P - p_q$, experimental set $E - p_q$, objective function f, and setting $p_q = s_{p_q j_q}$.

Our algorithm is intuitively simple to understand. Its goal at each step is to order the parameter and associated parameter setting by how broadly positive of an impact they have, marginalizing over all remaining parameters in the parameter space. It then sets the best parameter to its best setting and repeats the process with all remaining parameters until it hits the bottom and has to backtrack to the next best parameter setting. Given enough players, the algorithm will eventually test all experimental conditions.

$P - E$ can be thought of as the generalization space of the results; none of these parameters are directly set, but they are always randomly selected at each stage. The generalization power of most standard studies is often both implicit and minimal, in the effort to control as many variables as possible. But in our framework it is made explicit, and as we will see later can reasonably be quite large, since we can muster so many players.

The algorithm can be stopped at any point, giving the experimenter a partial experimental tree and the current-best node, B. A common stopping choice might be to have it stop once it reaches the bottom for the first time, resulting in the greedy selection of good parameter settings. In this case, if $k_{p_i} \leq K$ and $|E| = M$, the number of experiments the system will run is $O(KM^2)$. Unfortunately, if there are particularly nasty interactions between parameters, nothing short of a full search of the experiment parameter space and its associated $O(K^M)$ runtime is guaranteed to find the globally optimal setting. An easy solution is for the experimenter to combine parameters likely to interact into a single parameter with many settings. Or, if we are assume that there are no more than J-way interactions between parameters, we could allow the algorithm to explore all combinations of parameters of size J at each level, resulting in approximately $O(K^{\binom{M}{J}}(\frac{M}{J})^2)$ experiments. N,

the number of players assigned to each condition, must be chosen carefully to make good decisions in reasonable time.

FULL FRAMEWORK WITH IMPORTANCE SAMPLING

We are primarily interested in using games and online learning software. In these systems, we have the advantage that users are inexpensive, but the disadvantage that we may not have full control over our player sampling process. For example, game design constraints may make it difficult to give highly randomized interventions: a game with completely random levels may not be very fun to play. We also wish to be able to function in an offline setting in order to re-use existing datasets or if it is difficult to choose parameters for each new player in an online fashion.

We deal with both problems by extending our basic framework using importance sampling. Importance sampling is a commonly-used technique (ex. [14]) that allows us to estimate an expected function value from a desired distribution of the arguments, even though we can only sample from a different distribution of the arguments. This is accomplished by weighting our function evaluations. Specifically, let $f(x)$ be our objective function, $p(x)$ our desired distribution, and $q(x)$ our actual distribution. Then $E_p[f(x)] = E_p[f(x)\frac{q(x)}{q(x)}] = E_q[f(x)\frac{p(x)}{q(x)}]$. The last quantity is one we can estimate from data with $F_q = \frac{1}{N}\sum_{x \in X_q} f(x)\frac{p(x)}{q(x)}$, as long as we know both the sample and target distributions, and gives us an unbiased and consistent estimator. This technique allows us to run our full framework in offline situations with soft constraints on what interventions we can give to players, assuming that the dataset has non-zero probability for all possible settings of the experimental parameter set.

ASSESSMENT

We want to maximize player learning in our game. There are two challenges. The first is that players may quit at any time, so that an intervention may appear to be better just because it causes the least able players to quit. We deal with this by having short interventions and assessments, and assigning a score of 0 to players who quit before reaching the assessment. The second is that we need to be able to measure player knowledge. This is actually a major challenge: imagine the number of players we would lose by embedding a paper-and-pencil test in a free online game. In our implementation, we mitigate this problem by both embedding the test in the game itself, and only giving players a single question. For us, the resulting increase in noise of the objective function is offset by the large number of players we have. In other scenarios, longer tests may be a better choice.

While this is not a standard testing approach, we can do this because we are interested only in comparing expected assessment scores between different experimental conditions. At any particular stage, the population we are measuring, X_k consists of players who were directed into some particular condition C_k whose efficacy we wish to measure. The expected test score F_{C_k} for any player $x \in X_k$ for our randomized test is obtained by simply averaging over the test scores that we observe over all players, as long as players are sampled independently and identically distributed (from X). This approach bears some similarity to the one taken in domain sampling theory [11], one of the classical testing theories from psychometrics. We avoid many complications because we do not need to estimate single-user scores, only population-level ones, and our choice is justified because we are sampling directly from the population of interest.

To get a sense of how reliable our results are, we would like to establish confidence intervals for our assessment objective function. This is a non-trivial task, given that there is no a-priori knowledge of the objective function distribution, and we re-weight our samples with importance sampling. If we assume a stationary distribution of player scores, we can use a general resampling method known as bootstrapping [12], which repeatedly samples from our empirical data and calculates a test statistic on these resampled batches to estimate quantities relating to the original, unknown distribution. In our case, the test statistic is the mean, and we are interested in obtaining 95% confidence intervals of the mean. Since we have no guarantee of the symmetry of our sample mean around the true mean, we use the centered bootstrap percentile method [32]. Our framework does not depend on the method of calculating confidence intervals, however, so for certain classes of objective functions it may be considerably faster to calculate these intervals with closed-form solutions or more intelligent sampling methods.

EXAMPLE IMPLEMENTATION

Now that we have described our general automatic experimentation framework, we demonstrate its power with a full implementation in a specific setting, along with experimental results. Our platform is an educational game, with players gathered from a popular flash game website targeted at schoolchildren and teachers [9]. Taken together, our importance sampling method and our randomized assessments over populations will allow us to run the full system on a 2 x 2 x 4 x 4 experimental parameter space on a data set collected previously for a different use. This will allow us to discover what number line properties are most likely to lead to player near-transfer on a second, randomized test number line.

Treefrog Treasure

Treefrog Treasure is a platformer game that involves jumping through a jungle world and solving number line problems to reach an end goal. The player must navigate sticky, bouncy, and slippery surfaces and avoid hazardous lava to win successively more complex levels. Number line problems serve as barriers that the player must solve by hitting the correct target location, as shown in Figure 3. It has been played by over 5 million players worldwide.

Experimental Design

Our dataset was collected from June 3, 2013 to June 20, 2013. Players went through several tutorial levels before reaching the experiment levels. After cleaning our data, we had 34,197 players who made it past the tutorial.

We are trying to find the type of number line that leads to greatest player performance on a randomized test number

Figure 2. A screenshot of Treefrog Treasure, our source of users. Players navigate through a physics-based world, solving number line problems along the way. Notice that the number line has full tick marks, pie chart labels on the line, and a symbolic (ex. $\frac{a}{b}$) target representation. In our experiment, these are a few of the parameters we allow our system to automatically explore to determine which types of number lines lead to maximal near-transfer.

line. To do this, we consider each player as a sequence of many pairs of number lines, and treat each pair as an experimental unit. This gives us 361,738 pairs. This violates certain assumptions about the independence of variables in classical statistical tests, but greatly increases the amount of available data. We will strictly adhere to the correct assumptions when we validate our results on a new dataset, later, and we will see that our major results continue to hold.

We chose number lines in general as they are a popular pedagogical tool, and a fair amount of evidence exists suggesting that much whole and rational number knowledge is organized around a mental number line [6], [30]. Our experimental parameters alter the way number lines are presented, and can be seen in Table 1. We chose these parameters because they have all been the subject of previous research and are subject to varying amounts of controversy. Tick marks may allow students to find fractions directly through a double-counting method [22]. Hints are often considered necessary by educators [16], but can have negative effects when students "game" them [7]. Finally, there are many ways to represent fractions: pie charts and other area models, operators and linear models [27], magnitudes [30], standard symbolic notation, and so on. Showing multiple representations to students is widely thought to be useful, but may actually be worse than a single representation in some circumstances [29].

The parameter settings of the first number line in each pair constitutes the experimental condition. The full parameter set can be seen in Table 1. Our experimental set consists of Ticks, Animations, Backoff Hints, Target Representation, and Label Representation. We suspect that Target and Label representations are likely to interact, so we treat them as a single parameter, Representation, with four settings for Target and Label: Symbolic/Symbolic, Symbolic/Pie, Pie/Symbolic, Pie/Pie. This means that our results are meant to hold for different Fraction and Initial Labels values, which are always randomly chosen.

Figure 3. The animation condition on the left shows the player how to divide up the number line. The backoff condition in the middle fills in labels and eventually tells the player where to hit. The ticks condition on the right either divide up the number line into segments when ticks are present, or leave it empty besides the 0 and 1 labels when ticks are absent.

We show the results of our system on two objective functions.

1. *Correctness* 1.0 if the player answers the second number line correctly on the first try, 0.0 if they answer incorrectly or quit/restart before reaching it. This corresponds to fraction-placement ability.

2. *Persistence* 1.0 if the player eventually answers the second number line, 0.0 if they fail to answer it or quit/restart before reaching it. This corresponds to fraction-placement persistence.

Correcting sampling distributions

We want the number line parameter settings to be selected uniformly at random. Thus, if one experimental condition had a better objective function value than another, it would mean that some particular settings for the first number line (marginalized over the specific Fraction and Initial Label set) increased player performance across our randomized second number line.

Unfortunately, the dataset was collected for a different purpose, so the actual distribution is different than our distribution of interest. In this dataset, number lines are linked, two at a time. More specifically, the first two number lines always share the same value of Ticks, Animations, Backoff Hints, Target Representation, Label Representation, Initial Labels, and the Fraction denominator d. These are chosen uniformly at random. Then each number line in the pair has the Fraction numerator selected uniformly at random from 1 to $d-1$. Likewise, the second pair's parameters are selected (independently) using the same process, and so on.

This has several implications. The first is that certain fractions are over-represented relative to the desired uniform distribution over fractions. For example, $\frac{1}{2}$ is much more likely to be selected than $\frac{1}{9}$. The second is that half of our generated pairs will match on all parameters except the Fraction numerator due to the parameter pairing, and the other half will be independently and randomly generated from the process above.

As suggested above, we can use importance sampling to address these problems. This will up-weight the undersampled cross-pair links (second and third, fourth and fifth, etc.), and

Parameter	Settings	Interpretation
Fraction	Any $\frac{a}{b} \in (0,1), b \leq 9$	The target fraction the player must hit
Ticks	Present, Absent	For target $\frac{a}{b}$, we can display tick marks for each fraction $\frac{n}{b}$.
Animations	Present, Absent	If the player misses a target $\frac{a}{b}$, they might receive an lengthy pie chart animation showing how to divide up the number line into b parts.
Initial Labels	[0,1]	For target $\frac{a}{b}$, the proportion of labels of $\frac{n}{b}$ fractions shown at the start.
Backoff Hints	1, 2, 3, 4	The number of misses for target $\frac{a}{b}$ before the progressive hinting system fills in all labels for $\frac{n}{b}$ and displays the correct answer.
Target Representation	Symbolic, Pie	How the target fraction is displayed.
Label Representation	Symbolic, Pie	How fraction labels on the number line are displayed.

Table 1. The parameter space for our experiment.

down-weight the oversampled intra-pair links (first and second, third and fourth, etc.). For any experimental condition C_k, we have a set of parameters that are already set to some known value, and a set of parameters that should be uniformly random. This gives us the desired sampling distribution over the first number line. Since we also know that our desired objective function is some measure of performance on an independent, uniformly random second number line, this specifies the full desired distribution over pairs. But since we know the original distribution used to generate the data, we simply use importance sampling as above to reweight each objective function valuation in our dataset to calculate V_k, the expected player score under our desired distribution.

Results

Since we have at least one player in every experimental condition, it's possible to finish the depth-first search and generate the full experimental tree. However, even just the bottom of the tree contains 64 possible parameter setting combinations, making it difficult to show the full set of results. Instead, we show the parameters the algorithm greedily selected and its evaluation of the objective function for each of the different settings, stopping once it has set each parameter.

The results are shown in Figure 4. This is only a narrow, greedy slice of each experimental tree. At each stage, our algorithm finds the single parameter setting that maximizes the objective function, while averaging over all other parameters. It then sets this parameter to the best setting and repeats this process with the remaining parameters. Thus the Representation pie/pie setting is the broadest, best parameter setting among the entire experimental set in the *correctness* tree in Figure 4(a), the Backoff Hints 3 result is the broadest, best parameter setting only when the Representation is given to be pie/pie, and so on.

The confidence intervals given at each level of the tree grow wider as we go down. This is because data becomes sparser at each level, since we do not have control over the sampling distribution. In the basic online framework, the system would instead direct players to the condition in question, decreasing the amount of data near the top and increasing it near the bottom. Based on the amount of overlap present, we can guess that the results of the top and possibly second layer are reasonably trustworthy, but that we should be increasingly suspicious as we test more specific conditions.

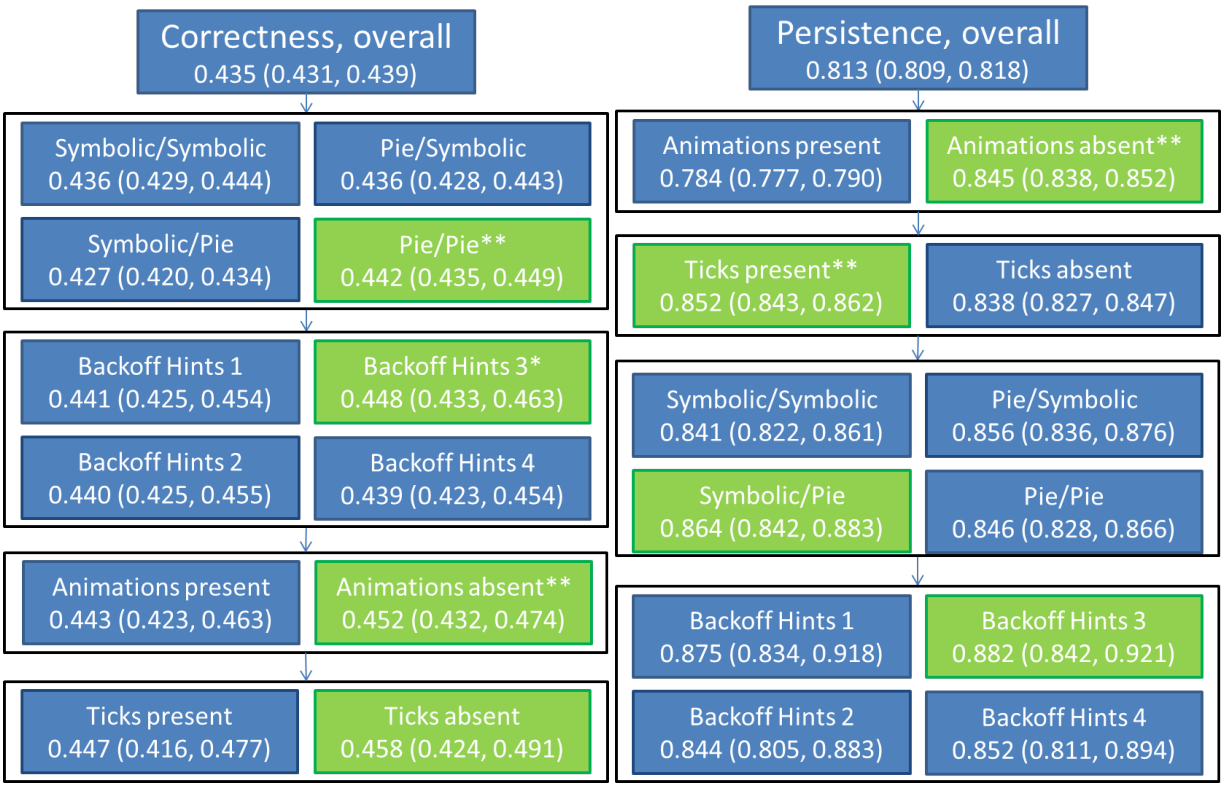
An important question is whether the greedy method finds reasonable settings, in practice. We can of course construct examples such that for any deterministic strategy, the algorithm must perform an exponential search to find the best settings. This can be quite bad for large parameter spaces, though it is probably unlikely in practice. Since our original dataset contains samples from all over the experimental tree, we can exhaustively search all possible experimental conditions at each depth to see when the greedy search diverges. Our greedy selection of the *correctness* diverges from the global optimum on the third and fourth levels with average score 0.460 and 0.474, respectively. These values are well-within the greedy selection's confidence intervals as seen in Figure 4(a). Our greedy selection of in the *persistence* condition finds the globally best selection at each level of the tree. Thus we conclude that greedy selection is reliable in this particular domain, especially at the top of the tree where data are plentiful and only a few parameters are set.

Validation

Our results so far could be useful to a game designer, with appropriate validation on new players to avoid overfitting. However, we would like even more: we want our framework to suggest important parameters and likely-effective settings for further experimentation, or even to generate research results outright. Unfortunately, our methodology makes a number of assumptions that make standard statistical tests inapplicable, and runs so many experiments that it is virtually guaranteed to find spurious results. We can, however, use our results to generate hypotheses that are testable on a new dataset.

We use a second dataset, again sampled according to the same distribution as the original. This dataset consists of 9,675 players of Treefrog Treasure from June 20, 2013 to July 9, 2013. We did not use this dataset to help us develop our system, hence it is similar to the final test dataset common in supervised machine learning.

Recall that the first two number lines share most parameters, including all of our experimental parameters, so they can serve as our experimental condition. All other parameters are chosen independently of the experimental ones, so the conditions are comparable. The third number line is itself chosen randomly and independently from the first two, and can serve as our assessment.



(a) Correctness (b) Persistence

Figure 4. A greedy slice of the experimental space explored, for two objective functions. Objective function evaluations are given with 95% confidence intervals, given by the centered bootstrap percentile method. Our algorithm conditions on more parameters as we go deeper down the tree, so that the results at the bottom have all experimental parameter values set according to their best observed settings. In addition, we will later statistically test the results on a separate dataset. * marks results that will be marginally significant, $p < 0.10$. ** marks results that will be significant, $p < 0.05$.

Thus, for any combination of parameter settings, we can ask whether players given number lines matching those criteria on the first two number lines performed better on the third than everyone else, a cross-sectional sampling scheme. We collect only one datapoint from each player, allowing us to meet the independence of samples criteria. Since both our objective functions are of the form "Pass" or "Fail", represented as 1.0 and 0.0 respectively, we use the χ^2 two sample test in each case. Other types of objective functions will in general have different appropriate tests, such as ANOVA or Mann-Whitney U.

Each experimental tree generates a large number of potential hypotheses; we will simply focus on the most basic ones, which is whether the chosen parameter settings lead to increasingly "good" outcomes as we go down the tree. Each comparison will be performance of players with the settings of the experimental parameters at that point, as compared to everyone else. The results are shown in Tables 2 and 3.

Remember that the wide confidence intervals at the deep ranges of the tree with many parameters set led us to suspect those results. Indeed, our statistical tests confirm this fact: we achieve significantly or marginally significantly better performance following our automatically-generated results at the top two layers, but mostly do not see significant effects on the bottom two. Thus our validation results are not surprising, and underscore the need for a validation set when running the system offline to avoid overfitting.

Finally, the effects may seem relatively weak, with a 2.4% increase from using the pie/pie representation on *correctness*. However, this is because we are measuring differences of interventions consisting of two numberlines and no explicit instruction. If a 10% difference in test scores after thirty minutes of instruction is good, then a 2% improvement after one minute may be reasonable. The extension of effective short interventions to effective long interventions is not trivial, and is left to future work.

DISCUSSION

Hypothesis Generation

Our primary purpose is to introduce an automatic experimentation framework. To demonstrate its utility, we have shown that we can use our implementation to discover interesting information and find potential educational hypotheses to further explore. We certainly do not claim that our findings are highly general, mature educational results. There are many caveats: the intervention is extremely short, the measured task is near-transfer onto a broadly randomized number line, the population is drawn from an online educational game, and so on.

Representation	Backoff Hints	Animations	Ticks	Mean	Other mean	Statistics
pie/pie	Any	Any	Any	0.431	0.407	$\chi^2(1, N = 9675) = 4.44, p = .0035$
pie/pie	3	Any	Any	0.447	0.410	$\chi^2(1, N = 9675) = 3.13, p = .077$
pie/pie	3	No	Any	0.471	0.411	$\chi^2(1, N = 9675) = 4.23, p = .040$
pie/pie	3	No	Yes	0.429	0.412	$\chi^2(1, N = 9675) = 0.15, p = .694$

Table 2. The results for correctness on the final validation set. The first line says that interventions with double pie charts are better than all others. The second line says that interventions with double pie charts and level 3 backoff hints are better than all others, and so on.

Animations	Ticks	Representation	Backoff Hints	Mean	Other mean	Statistics
No	Any	Any	Any	0.890	0.864	$\chi^2(1, N = 9675) = 14.18, p < .001$
No	Yes	Any	Any	0.896	0.870	$\chi^2(1, N = 9675) = 11.84, p < .001$
No	Yes	symbolic/pie	Any	0.892	0.876	$\chi^2(1, N = 9675) = 1.33, p = .249$
No	Yes	symbolic/pie	3	0.877	0.877	$\chi^2(1, N = 9675) < 0.001, p = 0.995$

Table 3. The results for persistence on the final validation set.

That being said, our results suggest broader hypotheses that could now be tested either in our framework, with lengthier interventions and more comprehensive assessments, or in a standard fashion in a school or lab setting. While the expert specified the parameter space, she did not need to decide particular parameter settings that were likely to perform better than others. This reduces our reliance on expert knowledge and makes it less likely that we will miss important results due to lack of extensive exploration.

As one example from our *correctness* results, we see that the pie/pie representation is significantly better than any other representation combination at improving player performance on a huge variety of number lines with both symbolic and pie chart representations for targets and labels. Educational experts that we spoke with found this to be quite interesting, since number lines almost always appear with symbolic notation. Not only is this a statistically significant result on an extremely rare representation combination that bears further research on its own, it also has potential implications for multiple representations research in general.

To explain further, the early math educational literature generally supports the notion of multiple representations in supporting learning [24], but only in certain circumstances. Many students have difficulty converting back and forth between different representations [31]. One of the reasons multiple representations may sometimes not be beneficial are that students simply opt to ignore presented number lines or informative diagrams when they are given with no added explanation [13]. In the fraction domain specifically, other researchers have found that multiple graphical representations may actually be harmful relative to single representations [1], unless accompanied by a self-explanation prompt.

Although our intervention number lines do offer hints, our number lines have no explanations nor prompts in the traditional sense. Yet using pie charts together with number lines lead to superior performance on the test line, compared to using number lines using the standard symbolic notation. Thus our system may have found an example where multiple representations are useful, without the additional explanation or support suggested by the literature.

We do not know why this is the case in our game, but one explanation might be that understanding symbolic notation may be more difficult than understanding pie charts, which at least are seen outside of the classroom. Then players who are not proficient with number lines may learn them faster or be more willing to play only when they can map them to a more familiar pie chart representation. The opposite possibility is that players have overfit in the classroom to number lines with symbolic notation. In this case, they would have difficulty answering the test number line questions that involve pie charts, and so the most profitable thing to practice would be the number line and pie chart combination. Though, we also note that pie/symbolic and symbolic/pie conditions are worse as well; perhaps the difficulty of mapping between three representations outweighs the potential benefits of seeing a pie chart target with a standard, symbolically-labeled number line.

Regardless of the explanation, our system was able to automatically find and run an interesting experiment that we would not have thought to try. The generated results were confirmed on a separate dataset, and differ in key ways with well-accepted literature, suggesting extensions to existing theories and further research to be done. This demonstrates the exploratory power of our method.

Finally, in this paper we have concentrated on parameters in an educational game; however, our method should be applicable to other domains, as well. For example, in the e-commerce domain, one could consider the parameter space of page layouts, checkout strategies, and item recommendation algorithm, with an objective function of clickthrough rate. Or a polling experiment on Mechanical Turk might ask which combinations of introduction, phrasing, question ordering result in the most consistent survey results. The key is to have a constant stream of users, and the ability to choose parameter settings for users and measure an outcome.

Limitations

Our work has important limitations. We wish to stress that the results are only strictly applicable to the user population they were generated from: in our case, players of our educational game. This can be mitigated in certain domains where demographics can be collected. When this is not possible, it may be best to treat the obtained results as hypotheses to test for future experiments on the desired population.

Furthermore, the algorithm is only as effective as the parameter space specified by the experimenter. It is entirely possible that the given parameters have negligible impacts on the objective function. In this case, the algorithm will greedily select parameter settings that appear very close to the global average, which may serve as a signal that a new parameter space should be devised. Researchers with a solid grasp of the underlying behavioral theories may be able to create more effective parameter spaces.

We also caution that problems arise in certain platforms, especially when users are not invested in the system. In the games domain, users can quit at any time; if a long intervention is desired, changes in the objective function may be caused by survivor bias induced by particular users leaving. As an example, an extremely difficult number line might appear to have a strong test score, but only because it caused all the players bad at answering number lines to quit. We control for this effect by having extremely short interventions so that the probability of quitting is low, and giving players who quit the lowest possible score. This protects us from spurious results caused by biased patterns of quitting, but also (intentionally) entangles learning and engagement. This issue is much less prominent in Mechanical Turk or software being used in schools by teachers, where the populations are more invested in finishing the intervention.

Also, this approach is focused only on exploring hypotheses related to the overall effects of system-controllable behaviors. Many factors such as age, gender, personality, performance, etc. are not directly controlled by the system, but are frequently studied in educational literature. For example, this system cannot identify different groups of people which need different interventions based on past performance. This type of useful adaptivity is challenging to achieve with limited data, and is left for future work.

FUTURE WORK

This is a new domain, only made possible in the past few years through the increasing use of the Internet. As such, there is a tremendous number of possible ways our framework could be extended or improved. We will list only a few of them.

Our implementation currently only handles standard, categorical factor experiments. This is not a fundamental limitation, but there is more work to be done to handle ordinal or numeric factors. Our system currently cannot deal with these variables because it does not know how to find the ideal parameter setting to use in a continuous range. One solution is to sample at random from numerical factors, then chop them into ranges that best separate the data as in regression trees. Another approach is to try using one of many well-known optimization techniques which attempt to find the best value, such as Covariance Matrix Adaptation.

As mentioned earlier, our search strategy in the space of experimental parameters is a staged, greedy selection designed for both maximizing the objective function value and finding which parameters are most important. This is appropriate in an educational domain. However, there are many other possible search strategies maximizing other goals. For example, a psychologist might care about variables causing the biggest difference in behavior, in which case a better strategy might be to choose parameters with greatest information gain, as often done in decision trees [28]. In this paper our results are taken from the offline case, where the search strategy is less important, but when players are committed to conditions online the search strategy is critical.

Future work includes investigating search strategies in the online case. If the researcher's interest is purely in mapping out the hypothesis space, one could imagine a search strategy that simply tries to find the most discriminative parameter at each level of the tree, using some well-known metric like information gain or Gini impurity. And departing from standard techniques, we could imagine a system that does a soft search over the parameter space to find the discriminative parts of the experimental tree. There are many online algorithms from the active learning and multi-arm bandit communities that attempt to do similar things, which can potentially be adapted to this framework.

CONCLUSION

Recent years have seen the emergence of large sources of user data. In this paper, we take advantage of these new data sources and propose a general, automated experimentation and hypothesis generation framework. This framework is specifically designed to automatically explore large hypothesis spaces in human behavioral research. Our importance sampling component allows the system to be used offline and when we have different distributions than the one of interest.

To show the usefulness of our framework, we implement it using an educational game. Using already-collected data, our system explores the hypothesis space for two alternative objective functions: maximizing player correctness and player persistence on a highly randomized test numberline. We find the most important parameters and their recommended settings and show that the greedy selection does a good job of finding the best settings at each level. We then confirm just a few of the most promising generated hypotheses on a already-collected, different data set. One of these hypotheses, generated from an unusual method of representing fractions on a number line, seems to be in opposition to recent work, which indicates that our system is indeed capable of automatically generating and testing interesting hypotheses that may not have been otherwise discovered.

ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship grant No. DGE-0718124, as well as the Office of Naval Research grant N00014-12-C-0158, the Bill and Melinda Gates Foundation

grant OPP1031488, the Hewlett Foundation grant 2012-8161, Adobe, and Microsoft.

REFERENCES

1. Ainsworth, S. E., Bibby, P. A., and Wood, D. J. Analysing the costs and benefits of multi-representational learning environments. *Learning with multiple representations* (1998), 120–134.
2. Andersen, E., Liu, Y.-E., Snider, R., Szeto, R., Cooper, S., and Popović, Z. On the harmfulness of secondary game objectives. In *FDG* (2011).
3. Andersen, E., Liu, Y.-E., Snider, R., Szeto, R., and Popović, Z. Placing a value on aesthetics in online casual games. In *CHI* (2011).
4. Andersen, E., O'Rourke, E., Liu, Y.-E., Snider, R., Lowdermilk, J., Truong, D., Cooper, S., and Popović, Z. The impact of tutorials on games of varying complexity. In *CHI* (2012).
5. Anderson, J. R., Corbett, A. T., Koedinger, K. R., and Pelletier, R. Cognitive tutors: Lessons learned. *The Journal of the Learning Sciences 4*, 2 (1995), 167–207.
6. Ansari, D. Effects of development and enculturation on number representation in the brain. *Nature Reviews Neuroscience 9*, 4 (2008), 278–291.
7. Arbreton, A. Student goal orientation and help-seeking strategy use.
8. Berman, F., Fox, G., and Hey, A. J. *Grid computing: making the global infrastructure a reality*, vol. 2. Wiley. com, 2003.
9. BrainPOP. http://www.brainpop.com/.
10. Buhrmester, M., Kwang, T., and Gosling, S. D. Amazon's mechanical turk: A new source of inexpensive, yet high-quality, data? *Perspectives on Psychological Science 6*, 1 (2011), 3–5.
11. Churchill, Gilbert A., J. A paradigm for developing better measures of marketing constructs. *Journal of Marketing Research 16*, 1 (1979), pp. 64–73.
12. Efron, B., and Tibshirani, R. *An introduction to the bootstrap*, vol. 57. CRC press, 1993.
13. Gagatsis, A., and Elia, I. The effects of different modes of representations on mathematical problem solving. In *IGPME*, vol. 2 (2004), 447–454.
14. Hastings, W. K. Monte carlo sampling methods using markov chains and their applications. *Biometrika 57*, 1 (1970), 97–109.
15. Henrich, J., Heine, S. J., and Norenzayan, A. The weirdest people in the world. *Behavioral and Brain Sciences 33*, 2-3 (2010), 61–83.
16. Hume, G., Michael, J., Rovick, A., and Evens, M. Hinting as a tactic in one-on-one tutoring. *Journal of the Learning Sciences 5*, 1 (1996), 23–49.
17. King, R. D., Rowland, J., Oliver, S. G., Young, M., Aubrey, W., Byrne, E., Liakata, M., Markham, M., Pir, P., Soldatova, L. N., Sparkes, A., Whelan, K. E., and Clare, A. The automation of science. *Science 324*, 5923 (2009), 85–89.
18. Kittur, A., Chi, E. H., and Suh, B. Crowdsourcing user studies with mechanical turk. In *CHI* (2008).
19. Kohavi, R., Crook, T., Longbotham, R., Frasca, B., Henne, R., Ferres, J. L., and Melamed, T. Online experimentation at microsoft. In *Third Workshop on Data Mining Case Studies and Practice Prize* (2009).
20. Kohavi, R., Henne, R. M., and Sommerfield, D. Practical guide to controlled experiments on the web: listen to your customers not to the hippo. In *KDD* (2007).
21. Langley, P. *Scientific discovery: Computational explorations of the creative processes*. MIT press, 1987.
22. Larson, C. N. Locating proper fractions on number lines: Effect of length and equivalence. *School Science and Mathematics 80*, 5 (1980), 423–428.
23. Lee, Y., Buchanan, B., and Aronis, J. Knowledge-based learning in exploratory science: Learning rules to predict rodent carcinogenicity. *Machine Learning 30*, 2-3 (1998), 217–240.
24. Lesh, R., Post, T., and Behr, M. Representations and translations among representations in mathematics learning and problem solving. *Problems of representation in the teaching and learning of mathematics* (1987), 33–40.
25. Li, L., Chu, W., Langford, J., and Schapire, R. E. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, ACM (New York, NY, USA, 2010), 661–670.
26. Lomas, D., Patel, K., Forlizzi, J. L., and Koedinger, K. R. Optimizing challenge in an educational game using large-scale design experiments. In *CHI* (2013).
27. Moss, J., and Case, R. Developing children's understanding of the rational numbers: A new model and an experimental curriculum. *Journal for Research in Mathematics Education 30*, 2 (1999), pp. 122–147.
28. Quinlan, J. R. *C4. 5: programs for machine learning*, vol. 1. Morgan kaufmann, 1993.
29. Rau, M. A., Aleven, V., and Rummel, N. Intelligent tutoring systems with multiple representations and self-explanation prompts support learning of fractions. In *AIED* (2009).
30. Siegler, R. S., Thompson, C. A., and Schneider, M. An integrated theory of whole number and fractions development. *Cognitive Psychology 62*, 4.
31. Sierpinska, A. On understanding the notion of function. *The concept of function: Aspects of epistemology and pedagogy 25* (1992), 23–58.
32. Singh, K., and Xie, M. Bootstrap: A statistical method, 2008.
33. Stamper, J. C., Lomas, D., Ching, D., Ritter, S., Koedinger, K. R., and Steinhart, J. The rise of the super experiment. In *EDM* (2012), 196–200.