Ascend by Evolv: Artificial Intelligence-Based Massively Multivariate Conversion Rate Optimization

Risto Miikkulainen, Myles Brundage, Jonathan Epstein, Tyler Foster, Babak Hodjat, Neil Iscoe, Jingbo Jiang, Diego Legrand, Sam Nazari, Xin Qiu, Michael Scharff, Cory Schoolland, Robert Severn, Aaron Shagrin

■ *Conversion rate optimization (CRO) means designing an e-commerce web interface so that as many users as possible take a desired action such as registering for an account, requesting a contact, or making a purchase. Such design is usually done by hand, evaluating one change at a time through A/B testing, evaluating all combinations of two or three variables through multivariate testing, or evaluating multiple variables independently. Traditional CRO is thus limited to a small fraction of the design space only, and often misses important interactions between the design variables. This article describes Ascend by Evolv,*[1] *an automatic CRO system that uses evolutionary search to discover effective web interfaces given a human-designed search space. Design candidates are evaluated in parallel online with real users, making it possible to discover and use interactions between the design elements that are difficult to identify otherwise. A commercial product since September 2016, Ascend has been applied to numerous web interfaces across industries and search space sizes, with up to fourfold improvements over human design. Ascend can therefore be seen as massively multivariate CRO made possible by artificial intelligence.*

In e-commerce, designing web interfaces (that is, web pages and interactions) that convert as many users as possible from casual browsers to paying customers is an important goal (Salehd and Shukairy 2011; Ash, Page, and Ginty 2012). While there are some well-known design principles, including simplicity and consistency, there are often also unexpected interactions between elements of the page that determine how well it converts. The same element, such as a headline, image, or testimonial, may work well in one context but not in others — it is often hard to predict the result, and even harder to decide how to improve a given page.

An entire subfield of information technology has emerged in this area, called conversion rate optimization, or conversion science. The standard method is A/B testing, that is, designing two different versions of the same page, showing them to different users, and collecting statistics on how well they each convert (Kohavi and Longbotham 2016). This process allows incorporating human knowledge about the domain and conversion optimization into the design, and then testing their effect. After observing the results, new designs can be compared and gradually improved. The A/B testing process is difficult and time-consuming. Only a very small fraction of page designs can be tested in this way, and subtle interactions in the design are likely to go unnoticed and unutilized.

An alternative to A/B is multivariate testing, where all value combinations of a few elements are tested at once. While this process captures interactions between these elements, only a very small number of elements is usually included (for example, 2–3); the rest of the design space remains unexplored. The Taguchi method (Rao et al. 2008; Kohavi and Thomke 2017) is a practical implementation of multivariate testing. It avoids the computational complexity of full multivariate testing by evaluating only orthogonal combinations of element values. Taguchi is the current state of the art in this area, included in commercial applications such as the Adobe Target.[2] However, it assumes that the effect of each element is independent of the others, which is unlikely to be true in web interface design. It may therefore miss interactions that have a significant effect on conversion rate.

This article describes an artificial intelligence (AI)-assisted technology for conversion optimization based on evolutionary computation. This technology is implemented in Ascend, a conversion optimization product by Evolv Technologies[1] (and formerly by Sentient Technologies[3]), deployed in numerous e-commerce websites of paying customers since September 2016. Ascend uses a customer-designed search space as a starting point. It consists of a list of elements on the web page that can be changed, and their possible alternative values, such as a header text, font, and color, background image, testimonial text, and content order. Ascend then automatically generates web-page candidates to be tested, and improves those candidates through evolutionary optimization.

Because e-commerce sites often have a high volume of traffic, fitness evaluations can be done live with a large number of real users in parallel. The evolutionary process in Ascend can thus be seen as a massively parallel version of interactive evolution, making it possible to optimize web designs in a few weeks. Intelligent traffic allocation through multiarmed bandit (MAB) methods can be used to identify best candidates reliably, and also to optimize overall performance over limited-duration campaigns. From the application point of view, Ascend is a novel method for massively multivariate optimization of web-page designs. Depending on the application, improvements of 20 to 200 percent over human design are routine using this approach (as listed in Table 2 below). These results are reliable across industries and search-space sizes.

This article describes the technology underlying Ascend, presents an example use case, an empirical comparison with the Taguchi method, and an extension to improved traffic allocation using MAB methods, summarizes the product status, and outlines future opportunities for evolutionary computation in optimizing e-commerce.

Background

With the explosive growth of e-commerce in recent years, entirely new areas of study have emerged. One of the main ones is conversion rate optimization, that is, the study of how web interfaces should be designed so that they are as effective as possible in converting users from casual browsers to actual customers. Conversion means taking a desired action on the web interface such as making a purchase, registering for a marketing list, or clicking on other desired links in an e-mail, website, or desktop, mobile, or social media application (Salehd and Shukairy 2011; Ash, Page, and Ginty 2012). Conversions are usually measured in number of clicks, but also in metrics such as resulting revenue or time spent on the site and rate of return to the site.

Conversions are currently optimized in a labor-intensive manual process that requires significant expertise. The web design expert or marketer first creates designs that s/he believes to be effective. These designs are then tested in an A/B testing process by directing user traffic to them and measuring how well they convert. If the conversion rates are statistically significantly different, the better design is adopted. This design can then be improved further, using domain expertise to change it, in another few rounds of creation and testing.

Conversion optimization is a fast-emerging component of e-commerce. In 2016, companies spent over $72 billion to drive customers to their websites (eMarketer 2016). Much of that investment does not result in sales — conversion rates are typically two to four percent (that is, 2–4 percent of the users that come to the site convert within 30 days). In 2014, only 18 percent of the top 10,000 e-commerce sites did any conversion optimization; in January 2017, 30 percent of them did so.[4] The growth is largely due to available conversion optimization tools, such as Optimizely,[5] Visual Website Optimizer,[6] Mixpanel,[7] and Adobe Target.[2] These tools make it possible to configure the designs easily, allocate users to them, record the results, and measure significance.

This process has several limitations: First, although the tools make the task of designing effective web interfaces easier, the design is still done by human experts. The tools thus provide support for confirming the experts' ideas, not helping them explore and discover novel designs. Second, because each step in the process requires statistical significance, only a few designs can be tested. Third, each improvement step amounts to one step in hill-climbing; such a process can get stuck in local maxima. Fourth, the process is aimed at reducing false positives and therefore increases false negatives, that is, designs with good ideas may be overlooked. Fifth, although the tools provide support for multivariate testing, in practice only a few combinations can be tested (for example, five possible values for two elements, or three possible values for three elements) — or, when using the Taguchi method, the variables are assumed to have independent effects. As a result, it is difficult to discover and use interactions between design elements.

Evolutionary optimization is well suited to address these limitations. Evolution is an efficient method for exploration; only weak statistical evidence is needed for progress; its stochastic nature avoids getting stuck in local maxima; and good ideas will gradually become more prevalent. Most importantly, evolution searches for effective interactions. For instance, Ascend may find that the button needs to be green, but only when it is transparent, and the header is in small font, and the header text is aligned. Such interactions are very difficult to find using A/B testing, requiring human insight into the results. Evolution makes this discovery process automatic. With Ascend, it is thus possible to optimize conversions better and at a larger scale than before.

Technically, Ascend is related to approaches to interactive evolution (Takagi 2001; Secretan et al. 2011) and crowdsourcing (Brabham 2013; Lehman and Miikkulainen 2013a) in that evaluations of candidates are done online by human users. The usual interactive evolution paradigm, however, employs a relatively small number of human evaluators, and their task is to select good candidates or evaluate the fitness of a pool of candidates explicitly. In contrast, in Ascend, a massive number of human users are interacting with the candidates, and fitness is derived from their actions (that is, convert or not) implicitly.

The Ascend Method

Ascend consists of defining the space of possible web interfaces, initializing the population with a good coverage of that space, estimating the performance of the candidates reliably, allocating traffic to candidates intelligently so that bad designs can be eliminated early, and testing candidates online in parallel. Each of these steps is described in more detail in this section.

Defining the Search Space

The starting point for Ascend is a search space defined by the web designer. Ascend can be configured to optimize a design of a single web-page, or a funnel consisting of multiple pages such as the landing page, selections, and a shopping cart. For each such space, the designer specifies the elements on that page and values that they can take. For instance, in the landing page example of figures 1 and 2, logo size, header image, button color, content order are such elements, and they can each take on two to four values.

Ascend searches for good designs in the space of possible combinations of these values. This space is combinatorial, and can be very large, for example, 1,100,000 (as used here). Interestingly, it is exactly this combinatorial nature that makes web-page optimization a good application for evolution. Although human designers have insight into what values to use, their combinations are difficult to predict, and need to be discovered by a search process such as evolution.

Initializing Evolution

A typical setup is that there is already a current design for the web interface, and the goal for Ascend is to improve over its performance. That is, the current design of the web interface is designated as the control, and improvement is measured compared with that particular design.

Because fitness is evaluated with real users, exploration incurs real cost to the customer. It is therefore important that the candidates perform reasonably well throughout evolution, and especially in the beginning.

If the initial population is generated randomly, many web interfaces would perform poorly. Instead, the initial population is created using the Control as a starting point. The candidates are created by changing the value of one element systematically. In a small search space, the initial population thus consists of all candidates with one difference from the control; in a large search space, the population is a sample of the set of such candidates. With such an initialization, most of the candidates perform similarly to the control. The candidates also cover the search dimensions well, thus forming a good starting point for evolution.

Estimating Performance

Ultimately, the fitness of a candidate is its conversion rate, that is, the ratio of people that convert to the total visitor of the web page. Because there is only a limited amount of traffic available to test each candidate, this rate is always a noisy estimate. However, it can be made more reliable in two ways: First, by taking a Bayesian prior into account (the conversion rate is unlikely to be arbitrary, but instead is likely to be similar to those of other candidates), and second, by estimating how likely the candidate's conversion rate is to be better than that of the control.

A prior estimate of the conversion rate can be obtained as the average of all candidates tested so far. A probability distribution of conversion rate is then built for the control and the candidate as demonstrated in figure 3. The proportion of area under the curve of candidate conversion rate distribution where it beats that of control is computed as the

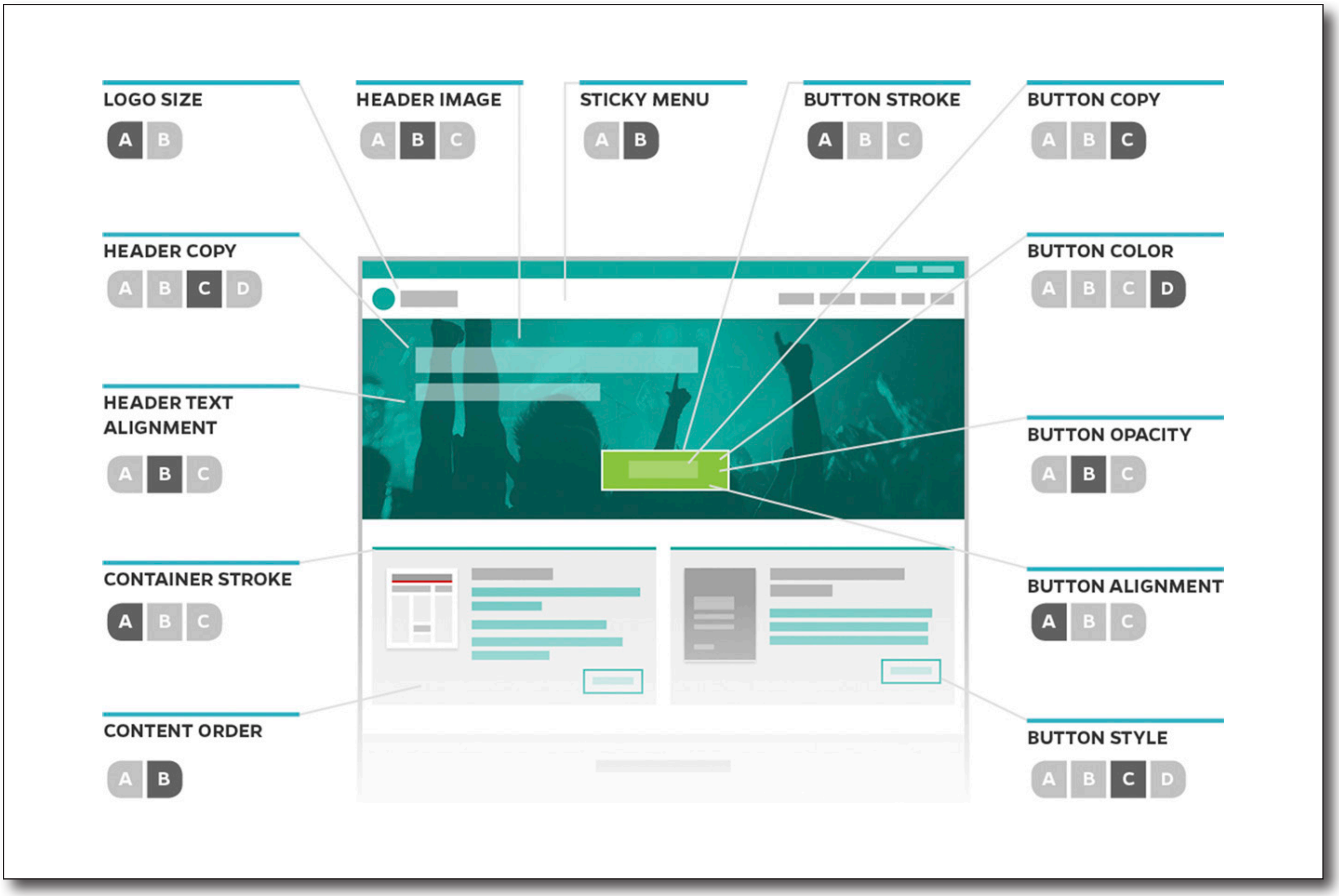


Figure 1. Elements and Values of an Example Web Page Design.

In this example, 13 elements each have two to four possible values, resulting in 1,100,000 combinations.

probability to beat control. This probability is then used as the fitness for the candidate.

While probability to beat control is a common technique in CRO,[8,9,10] the evolutionary optimization context in Ascend makes it possible to improve it further. Instead of computing the prior based on all candidates, it can be computed based on the candidates' evolutionary parents. They are most similar to the candidate, resulting in a more accurate prior, and therefore more reliable estimates.

Evolutionary Process

Each web page is represented as a genome, as shown for two example pages in figure 2 (columns 2 and 3). The usual genetic operations of crossover (recombination of the elements in the two genomes; columns 4 and 5) and mutation (randomly changing one element in the offspring; column 6) are then performed to create new candidates. In the current implementation, fitness-proportionate selection is used to generate offspring candidates from the current population. From the current population of *n* candidates, another *n* new candidates are generated in this way.

Because evaluations are expensive, consuming traffic for which most customers have to pay, it is useful to minimize them during evolution. Each page needs to be tested only to the extent that it is possible to decide whether it is promising, that is, whether it should serve as a parent in the next generation, or should be discarded. A process similar to age-layering (Hodjat and Shahrzad 2013; Shahrzad, Hodjat, and Miikkulainen 2016) is therefore used to allocate fitness evaluations. At each generation, each new candidate and each old candidate is evaluated with a small number (a maturity age) of user interactions, such as 2,000. The top *n* candidates are retained, and the bottom *n* discarded. In this manner, bad candidates are eliminated quickly. Good candidates receive progressively more evaluations, and the confidence in their fitness estimate increases.

In this process, Ascend learns which combinations of elements are effective, and gradually focuses the search around the most promising designs. It is thus sufficient to test only a tiny fraction of the search space to find the best ones, that is, thousands of pages instead of millions or billions.

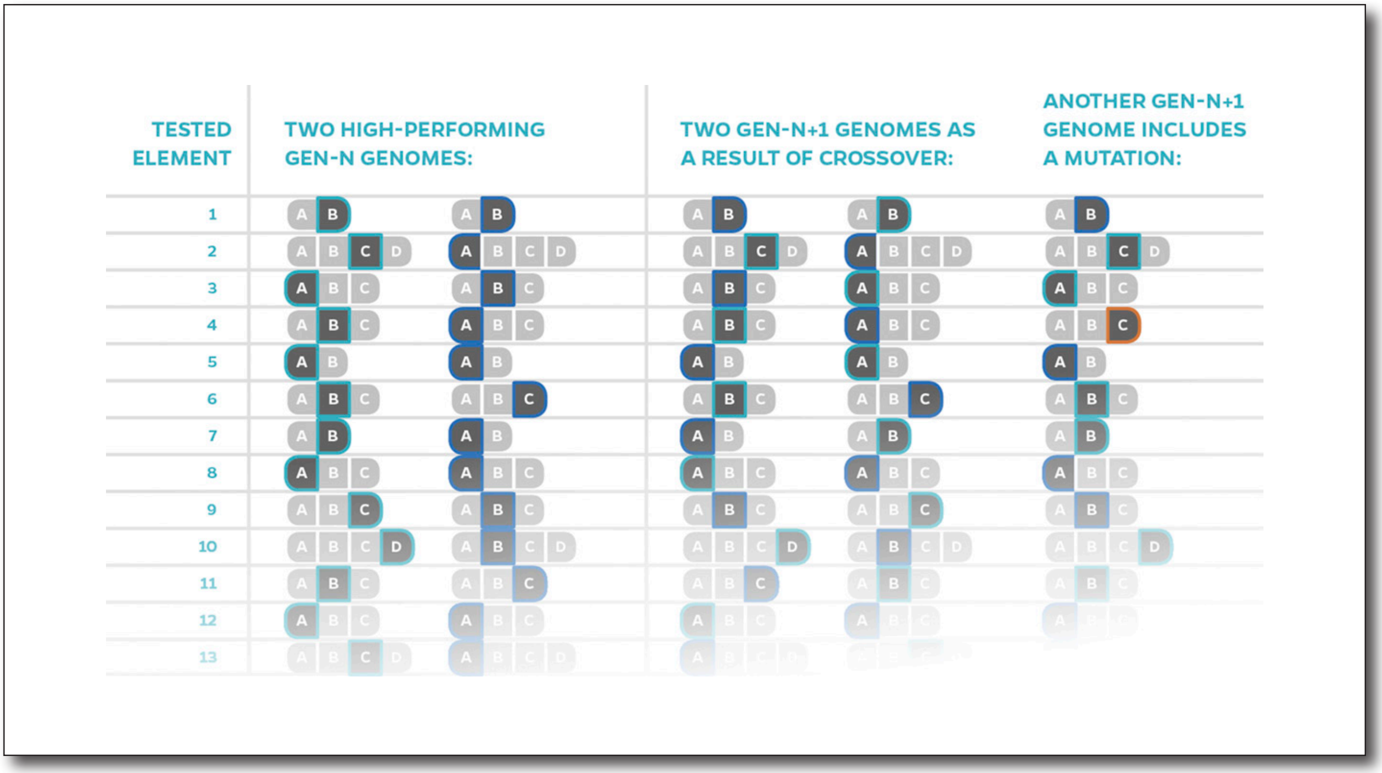

TESTED ELEMENT
TWO HIGH-PERFORMING GEN-N GENOMES:
TWO GEN-N+1 GENOMES AS A RESULT OF CROSSOVER:
ANOTHER GEN-N+1 GENOME INCLUDES A MUTATION:
1
2
3
4
5
6
7
8
9
10
11
12
13
A
B
C
D

Figure 2. Genetic Encoding and Operations on Web Interface Candidates.

Web pages are represented as concatenations of their element values with one-hot encoding. Crossover and mutation operate on these vectors as usual, creating new combinations of values.

Online Evolution

Whereas in simple cases (where the space of possible designs is small) such optimization can potentially be carried out by simpler mechanisms such as systematic search, hill-climbing, or reinforcement learning, the population-based approach is particularly effective because the evaluations can be done in parallel. The entire population can be tested at once, as different users interact with the site simultaneously. It is also unnecessary to test each design to statistical significance; only weak statistical evidence is sufficient to proceed in the search. In this process, thousands of page designs can be tested in a short time, which is impossible through A/B or multivariate testing.

Figure 4 shows the overall architecture of the system. A population of alternative designs (center) are adapted (right) based on evaluations with actual users (left). The population of designs (center) are evaluated with many users in parallel (left). The evolutionary process (right) generates new designs, and outputs the best design in the end. The system also keeps track of which design has been shown to which user, so that they get to see the same design if they return within a certain time limit (for example, the same day).

Case Study

As an example of how Ascend works, let us consider a case study on optimizing the web interface for a media site that connects users to online education programs. This experiment was run in September through November 2016 on the desktop traffic of the site.[11]

The initial design for this page is shown in the left side of figure 5. It had been hand-designed using standard tools such as Optimizely. Its conversion rate during the time of the experiment was found to be 5.61 percent, which is typical of such web interfaces. Based on this page, the web designers came up with nine elements, with two to nine values each, resulting in 381,024 potential combinations (figure 6). While much larger search spaces are possible, this example represents a midsize space common with many current sites.

The initial population of thirty-seven candidates was formed by systematically replacing each of the values in the control page with one of the alternative values, as described in the "Initializing Evolution" section. Evolution was then run for sixty days, or four generations, altogether testing 111 candidates with 599,008 user interactions total. The estimated conversion rates of the candidates over this time are shown in figure 7. This figure demonstrates that evolution was successful in discovering significantly better candidates than control.

As an independent verification, the three top candidates in figure 5 were then subjected to an A/B test using Optimizely. In about 6,500 user interactions, the best candidate was confirmed to increase the conversion rate by 43.5 percent with greater than ninety-nine percent significance (and the other two by 37.1 percent and 28.2 percent) — which is an excellent result given that the control was a candidate that was already hand-optimized using state-of-the art tools.

Unlike Control, the top candidates use bright background colors to draw attention to the widget. There is an important interaction between the background and the blue banner (whose color was fixed). In the best two designs (in the middle), the background is distinct from the banner, but not competing with it. Moreover, given the colored background, a white button with black text provided the most clear call for action. It is difficult to recognize such interactions ahead of time, yet evolution discovered them early on, and many of the later candidates built on them. Other factors such as an active call to action (that is, "Get Started" and "Find my Program" rather than "Request Info") amplified it further. At the time evolution was turned off, better designs were still being discovered — suggesting that a more prolonged evolution and a larger search space (for example, including banner color and other choices) could have improved the results further.

It is also interesting to note that during the experiment, the human designers referred to Ascend as "the ugly widget generator," suggesting that its designs were different from typical human designs. Remarkably, in doing so Ascend succeeded in creating a sense of urgency that is missing from the control design (figure 8), suggesting that Ascend can discover effective design principles of its own.

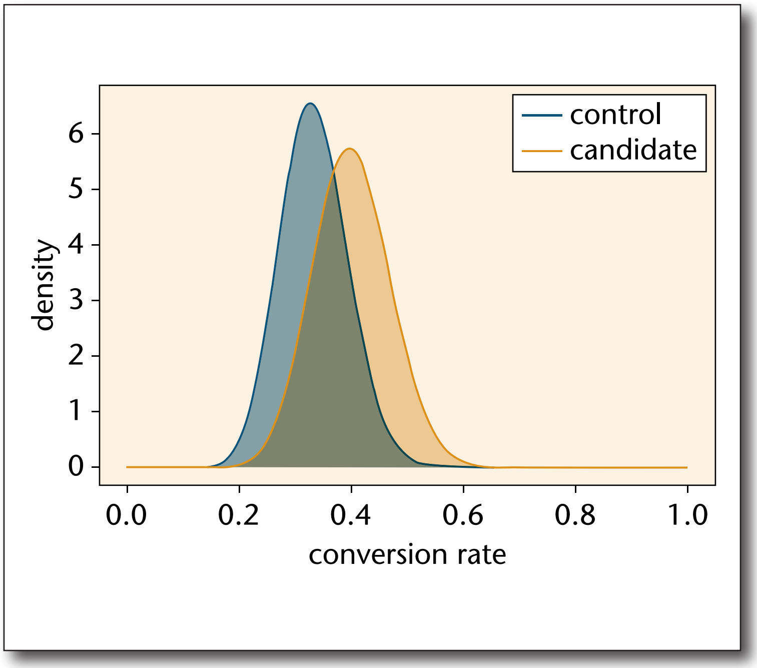


Figure 3. Probability Distribution of Control and Target Candidate Conversion Rates.

The area under the candidate curve that is above the control curve stands for the probability to beat control. Using that probability as the measure of performance instead of the estimated conversion rate leads to more reliable results. This technique is further enhanced in Ascend because the candidates are constructed through evolution. The parents' conversion rates provide a more accurate estimate of the prior than can be obtained otherwise.

Comparison with Multivariate Testing

The case study and numerous other examples reviewed in the "Discussion" section show that evolutionary optimization in Ascend discovers effective solutions. But does it offer improvement over other automated methods such as multivariate testing, and in particular the Taguchi method? Its ability to take advantage of interactions between design variables should allow it to find better designs than Taguchi. On the other hand, if variables are indeed independent, Taguchi might be a better method. A simulation study in this section is presented to test this hypothesis; for more details, see Jiang et al. (2018).

Simulation Setup

To study this question systematically, a simulated environment was created where the degree of interactions could be controlled. In the simulation, an evaluator is first constructed to calculate a candidate's true conversion rate based on the values it specifies for each variable. Simulated traffic is distributed to candidates and conversions are assigned probabilistically based on candidates' true conversion rate. The observed conversion rates are then used as the scores of the candidates in Taguchi and evolution methods. By setting the parameters of the simulation differently, different kinds of evaluators, that is, functions that determine the conversion rate CR(c) of candidate c, can be defined. For instance, the a simple linear evaluator is based on only bias W^0 (that is, the control conversion rate) and weight $W_i^1(c)$ for each individual variable i:

$$\mathrm{CR}(c) = W^0 + \sum_{i=1}^{n} W_i^1(c) \qquad (1)$$

The bias represents the conversion rate of the control candidate; the different choices for each variable add or subtract from the control rate. A nonlinear evaluator, in addition, takes interactions between variables into account:

$$\mathrm{CR}(c) = W^0 + \sum_{i=1}^{n} W_i^1(c) + \sum_{j=1}^{n} \sum_{k=j+1}^{n} W_{j,k}^2(c) \qquad (2)$$

That is, in addition to the bias and the individual variable contributions, it includes contributions $W_{j,k}^2(c)$ for each pair of variables j,k.

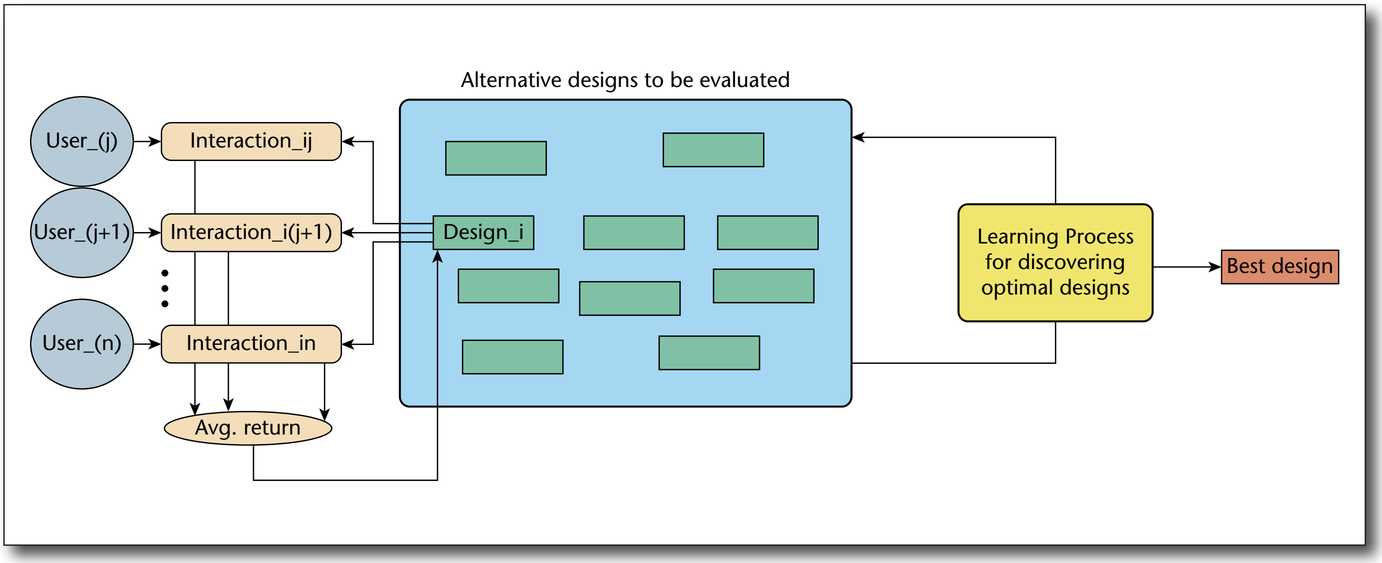
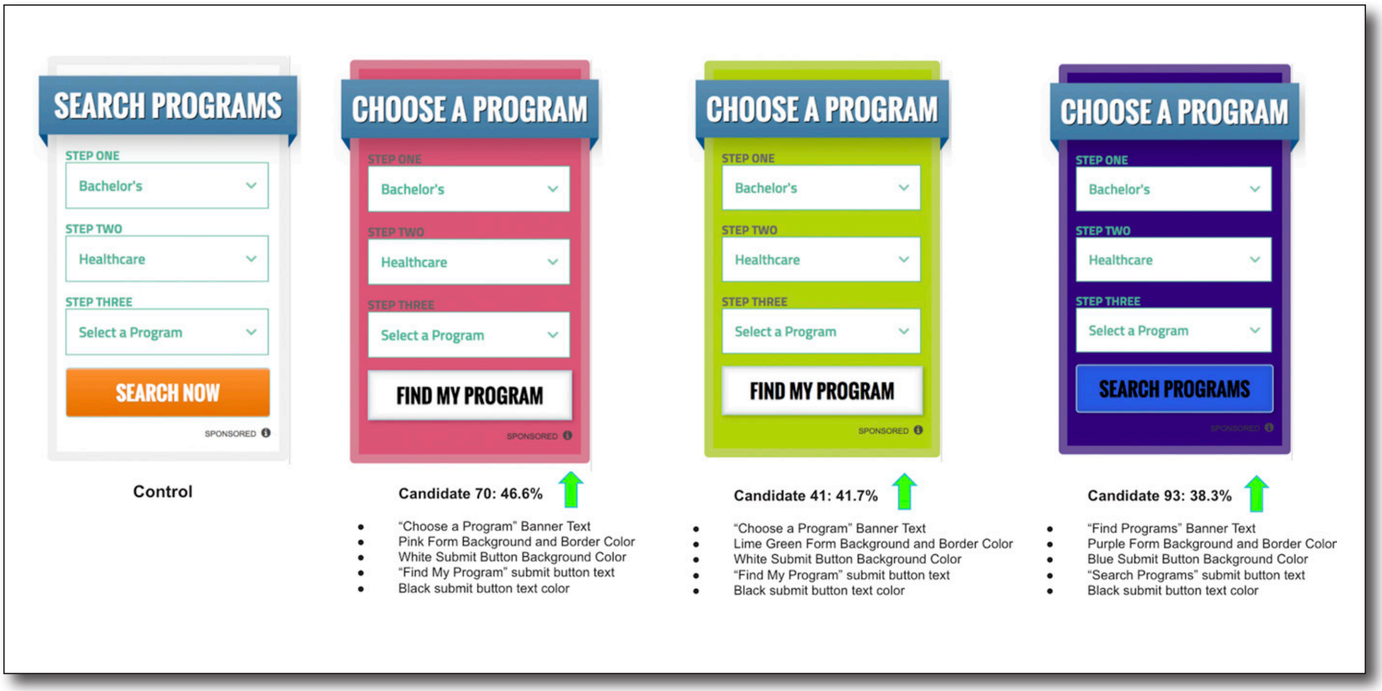


Figure 4. Overall Architecture of the Online Evolution System.

The outcome of each interaction (that is, whether the user converted or not) constitutes one evaluation of a design. Many such evaluations *ij* are run in parallel with different users *j* and averaged to estimate how good the design *i* is. After all designs have been evaluated, the adaptation process discards bad designs and generates more variations of the best designs. This process of generation, testing, and selection is repeated until a sufficiently good design has been found or the time allocated for the process has been spent. The best design found so far is output as the result of the learning process. The system thus discovers good designs for web interfaces through live online testing.



Control

Candidate 70: 46.6%

- "Choose a Program" Banner Text
- Pink Form Background and Border Color
- White Submit Button Background Color
- "Find My Program" submit button text
- Black submit button text color

Candidate 41: 41.7%

- "Choose a Program" Banner Text
- Lime Green Form Background and Border Color
- White Submit Button Background Color
- "Find My Program" submit button text
- Black submit button text color

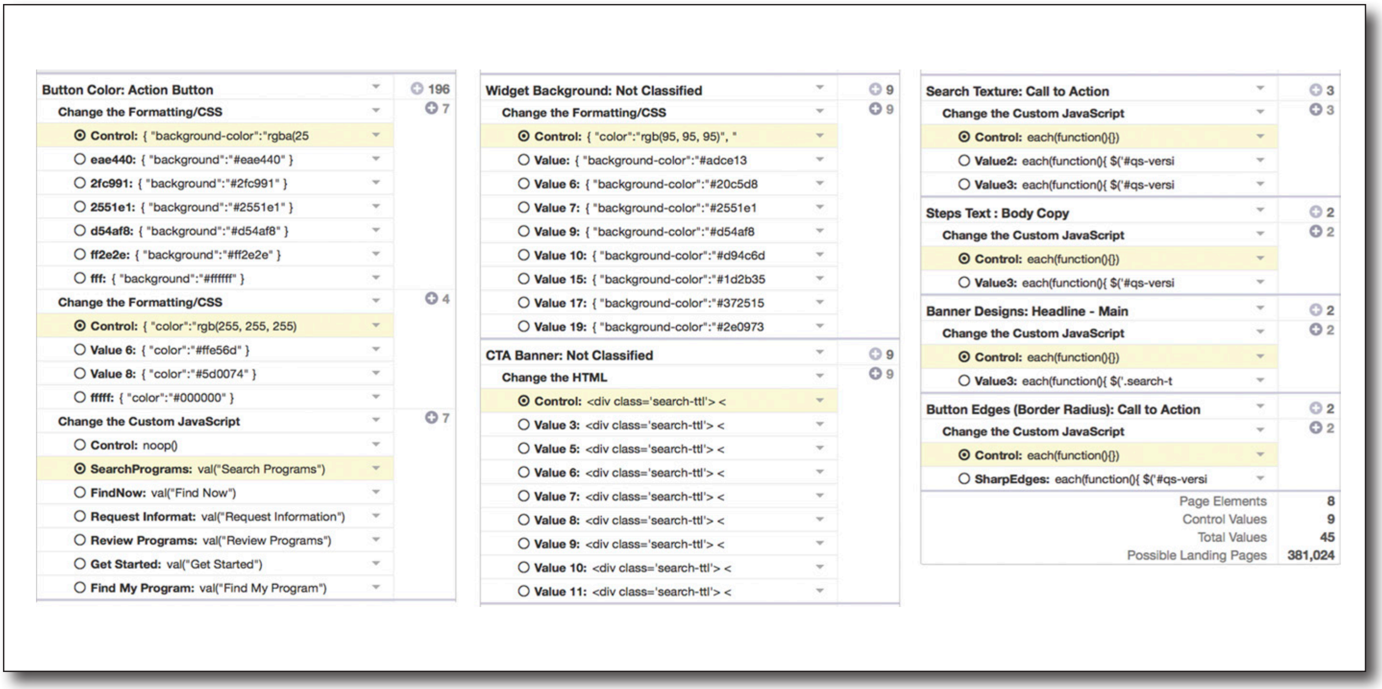
Candidate 93: 38.3%

- "Find Programs" Banner Text
- Purple Form Background and Border Color
- Blue Submit Button Background Color
- "Search Programs" submit button text
- Black submit button text color

Figure 5. The Control Design and Three Best Evolved Designs.

After 60 days of evolution with 599,008 user interactions, a design for the search widget was found that converted 46.6 percent better than the control (5.61 percent versus 8.22 percent), as well as other good designs. Much of the improvement was based on discovering a combination of colors that draws attention to the widget and makes the call to action clear.

Both the Taguchi candidates and the evolution candidates are represented in the same way — as concatenations of one-hot vectors representing the values for each variable in the Taguchi method, and actions for each gene in evolution. The total traffic for the Taguchi method and evolution algorithm is set to be equal, distributed evenly to all Taguchi candidates, but differently for evolution candidates



```
Button Color: Action Button                                   196
  Change the Formatting/CSS                                     7
    ⦿ Control: { "background-color":"rgba(25
    ○ eae440: { "background":"#eae440" }
    ○ 2fc991: { "background":"#2fc991" }
    ○ 2551e1: { "background":"#2551e1" }
    ○ d54af8: { "background":"#d54af8" }
    ○ ff2e2e: { "background":"#ff2e2e" }
    ○ fff: { "background":"#ffffff" }
  Change the Formatting/CSS                                     4
    ⦿ Control: { "color":"rgb(255, 255, 255)
    ○ Value 6: { "color":"#ffe56d" }
    ○ Value 8: { "color":"#5d0074" }
    ○ fffff: { "color":"#000000" }
  Change the Custom JavaScript                                  7
    ○ Control: noop()
    ⦿ SearchPrograms: val("Search Programs")
    ○ FindNow: val("Find Now")
    ○ Request Informat: val("Request Information")
    ○ Review Programs: val("Review Programs")
    ○ Get Started: val("Get Started")
    ○ Find My Program: val("Find My Program")

Widget Background: Not Classified                               9
  Change the Formatting/CSS                                     9
    ⦿ Control: { "color":"rgb(95, 95, 95)", "
    ○ Value: { "background-color":"#adce13
    ○ Value 6: { "background-color":"#20c5d8
    ○ Value 7: { "background-color":"#2551e1
    ○ Value 9: { "background-color":"#d54af8
    ○ Value 10: { "background-color":"#d94c6d
    ○ Value 15: { "background-color":"#1d2b35
    ○ Value 17: { "background-color":"#372515
    ○ Value 19: { "background-color":"#2e0973
CTA Banner: Not Classified                                      9
  Change the HTML                                               9
    ⦿ Control: <div class='search-ttl'> <
    ○ Value 3: <div class='search-ttl'> <
    ○ Value 5: <div class='search-ttl'> <
    ○ Value 6: <div class='search-ttl'> <
    ○ Value 7: <div class='search-ttl'> <
    ○ Value 8: <div class='search-ttl'> <
    ○ Value 9: <div class='search-ttl'> <
    ○ Value 10: <div class='search-ttl'> <
    ○ Value 11: <div class='search-ttl'> <

Search Texture: Call to Action                                  3
  Change the Custom JavaScript                                  3
    ⦿ Control: each(function(){})
    ○ Value2: each(function(){ $('#qs-versi
    ○ Value3: each(function(){ $('#qs-versi
Steps Text : Body Copy                                          2
  Change the Custom JavaScript                                  2
    ⦿ Control: each(function(){})
    ○ Value3: each(function(){ $('#qs-versi
Banner Designs: Headline - Main                                 2
  Change the Custom JavaScript                                  2
    ⦿ Control: each(function(){})
    ○ Value3: each(function(){ $('.search-t
Button Edges (Border Radius): Call to Action                    2
  Change the Custom JavaScript                                  2
    ⦿ Control: each(function(){})
    ○ SharpEdges: each(function(){ $('#qs-versi

                                        Page Elements           8
                                       Control Values           9
                                         Total Values          45
                               Possible Landing Pages     381,024
```

Figure 6. A Screenshot of the User Interface for Designing Ascend Experiments.

The screenshot shows the elements and values in the education program case study. Nine elements with two to nine different values each result in 381,024 potential web page designs; the first value in each element is designated as the control. This is a midsize problem typical of current web interface designs.

based on how many generations they survive. Eight generations of evolution were run with mutation rate = 0.01, elite percentage = 20 percent, and control conversion rate $W^0 = 0.05$.

The Taguchi Method

While full multivariate analysis would require testing all K^N combinations of N variables with K values each, the Taguchi method specifies a small subset of combinations to test using orthogonal arrays. A Taguchi orthogonal array is a matrix where each column corresponds to a variable and each row to a candidate to test. Each value represents the setting for a given variable and experiment. It has the following properties: The dot product between any two normalized column vectors is zero; and for every variable column, each value appears the same amount of times.

There are multiple ways of creating orthogonal arrays (Brouwer, Cohen, and Nguyen 2006; Hedayat, Sloane, and Stufken 2018). Table 1 shows an example of an orthogonal array of nine combinations, resulting from testing four variables of three values each.

To compute the effect of a specific variable value, the performance scores of the candidates corresponding to combinations for that value setting are averaged. Because all values of the other variables are tested an equal amount of times in an orthogonal array, their effects cancel out, assuming each variable is independent (Hedayat, Sloane, and Stufken 2018). For example, to compute the effect of value 2 of variable 3 in table 1, the scores of candidates 2, 4, and 9 are averaged. Similarly, for value 1, the scores of candidates 3, 5, and 7 are averaged. In a Taguchi experiment, all the candidates (rows) in the orthogonal table are tested, and the scores for candidates that share the same value for each variable are averaged in this manner. The prediction for the best-performing combination can then be constructed by selecting, for each variable, the value with the best such average score.

The Taguchi method is a practical approximation of factorial testing. However, the averaging steps assume that the effects of each variable are independent, which may or may not hold in real-world experiments. In contrast, population-based search makes no such assumptions. The simulations are designed to evaluation how the two approaches compare with different amounts of traffic and degrees of interactions.

Experimental Results

Three experiments were run comparing the Taguchi method with evolutionary optimization. In the first two, the goal was to find good candidates by the end of the experiment. In the first one, the variables had independent effects, and in the second, there were significant dependencies between pairs of variables. In the third experiment, the performance during the experiment was compared.

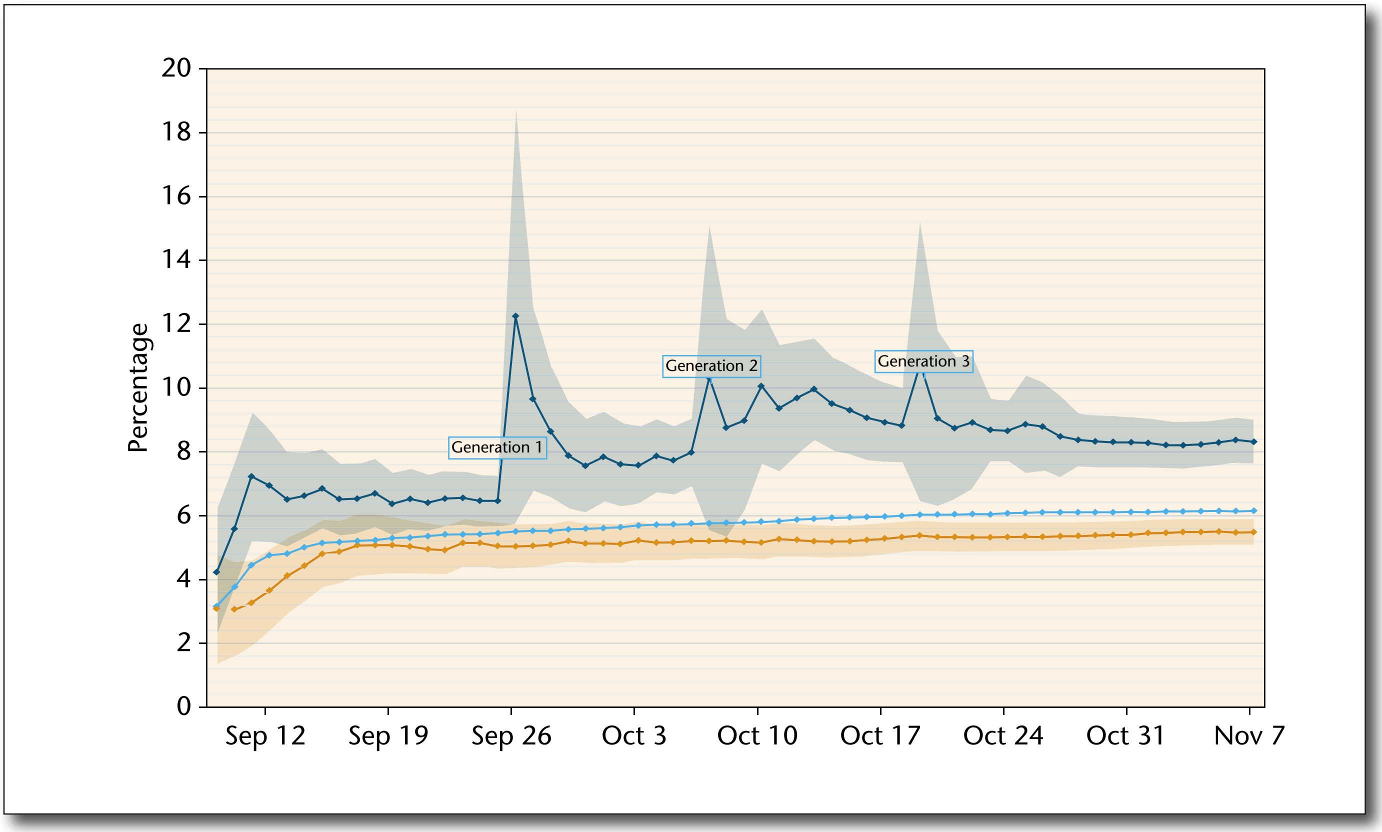


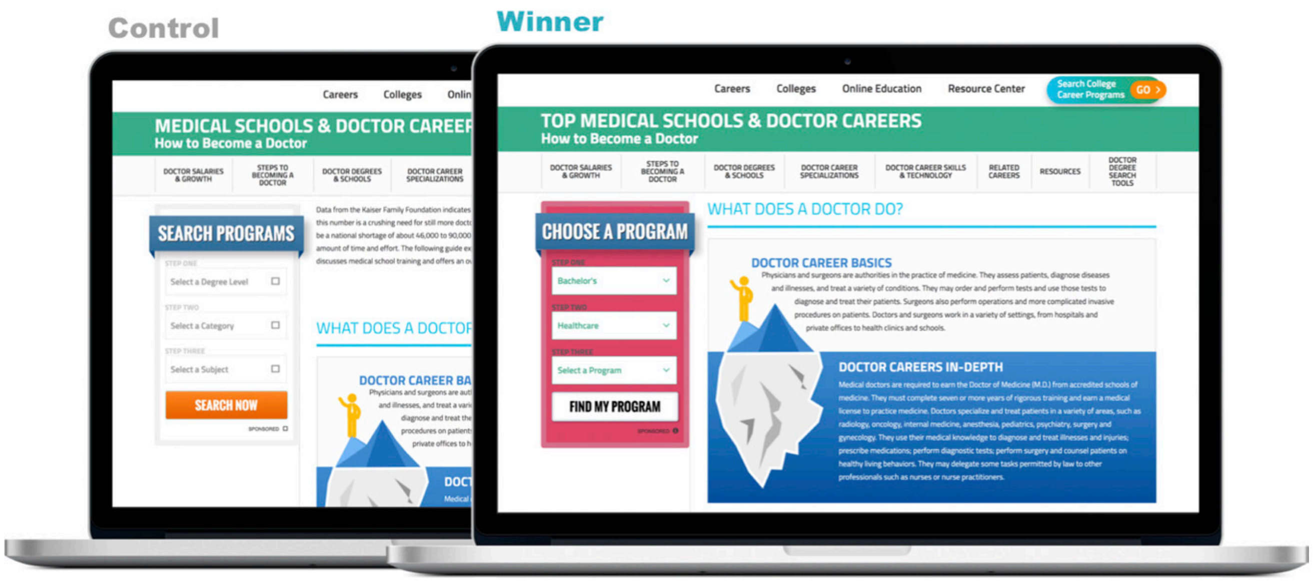
Figure 7. Estimated Conversion Rates through the 60-Day Online Evolution Run.

Days are in the *x* axis and the conversion rate on the *y* axis. The dark blue dots (on top) indicate the current best candidate, the light blue dots (in the middle) an average of all currently active candidates, and the orange dots (at the bottom) the estimated performance of the control design. The shaded areas display the 95 percent confidence intervals (from the binomial distribution with the observed mean). The blue peaks indicate the start of each new generation. Such peaks emerge because during the first few days, the new candidates have been evaluated only a small number of times, and some of them have very high estimated rates through random chance. Eventually they will be evaluated in a maturity age of 2,000 user interactions, and the estimates become lower and the confidence intervals narrower. The elite candidates are tested across several generations (as described in the "Evolutionary Process" section), resulting in very narrow intervals toward the end. Estimated conversion rates of the best candidates in later generations are significantly higher than control, suggesting that evolution is effective in discovering better candidates. Interestingly, the active population average is also higher than control, indicating that the experiment did not incur any cost in performance.

The first experiment uses a linear evaluator of equation 1 that assumes all changes are independent, and a simple genome that results in a small Taguchi array. These are the ideal conditions for the Taguchi method, and it is expected to perform well. The best settings for the Taguchi method are those with uniform numbers of values across all variables. In the experiment, four variables were used with three values each, that is, [3, 3, 3, 3], with $3^4 = 81$ combinations, resulting in nine rows in the orthogonal array.[12]

In this experiment, the true conversion rate for the best evolution candidate is steady at 0.0565 at all levels of traffic from 50,000 to 10,000,000 samples. The best predicted Taguchi candidate's true conversion rate is significantly lower, 0.0548, with low traffic, but eventually catches up as traffic increases to about 1,000,000 samples. It is also better than the best candidate in the actual Taguchi array, whose true conversion rate was approximately 0.0548 at all levels of traffic. Thus, under ideal conditions for Taguchi, both methods find equally good solutions given enough traffic. With low traffic, however, the evolutionary approach performs significantly better. The likely reason is that while in the Taguchi method the set of candidates is fixed, in evolution it is not. Evolution discards bad candidates quickly and does not spend much traffic on them; instead, it generates new candidates, and thus uses the traffic on evaluating increasingly better candidates.

In the second experiment, the nonlinear evaluator of equation 2 is used to simulate interactions that are likely to exist in the real world. Also, more variables with a varying number of possible values, that is, [3, 6, 2, 3, 6, 2, 2, 6], were used to make the problem more realistic. Figure 9 shows that in this case, the best predicted Taguchi candidate's true conversion rate is no longer comparable with evolution's. Furthermore, it does not even significantly outperform



Data from the Kaiser Family Foundation indicates
this number is a crushing need for still more docto
be a national shortage of about 46,000 to 90,000
amount of time and effort. The following guide ex
discusses medical school training and offers an o

Physicians and surgeons are auth
and illnesses, and treat a vari
diagnose and treat the
procedures on patient
private offices to h

Medical

Physicians and surgeons are authorities in the practice of medicine. They assess patients, diagnose diseases and illnesses, and treat a variety of conditions. They may order and perform tests and use those tests to diagnose and treat their patients. Surgeons also perform operations and more complicated invasive procedures on patients. Doctors and surgeons work in a variety of settings, from hospitals and private offices to health clinics and schools.

Medical doctors are required to earn the Doctor of Medicine (M.D.) from accredited schools of medicine. They must complete seven or more years of rigorous training and earn a medical license to practice medicine. Doctors specialize and treat patients in a variety of areas, such as radiology, oncology, internal medicine, anesthesia, pediatrics, psychiatry, surgery and gynecology. They use their medical knowledge to diagnose and treat illnesses and injuries; prescribe medications; perform diagnostic tests; perform surgery and counsel patients on healthy living behaviors. They may delegate some tasks permitted by law to other professionals such as nurses or nurse practitioners.

Figure 8. Comparison of the Evolved Widget with the Control.

In an independent A/B test, the winning design (on the right) was found to convert 43.5 percent better than the control. Ascend discovered a way of making the call to action more urgent, demonstrating that it can come up with principled, effective solutions that human designers may overlook.

	Variable 1	Variable 2	Variable 3	Variable 4	Performance
Combination 1	0	0	0	0	p1
Combination 2	0	1	2	1	p2
Combination 3	0	2	1	2	p3
Combination 4	1	0	2	2	p4
Combination 5	1	1	1	0	p5
Combination 6	1	2	0	1	p6
Combination 7	2	0	1	1	p7
Combination 8	2	1	0	2	p8
Combination 9	2	2	2	0	p9

Table 1. Example Taguchi Array of Four Variables with Three Levels Each.

its best tested candidate. Interestingly, the performance of the evolutionary algorithm is not significantly worse with interacting versus independent variables, demonstrating its ability to adapt to complicated real-world circumstances.

While the main goal in conversion optimization is to find good candidates that can be deployed after the experiment, in many cases it is also important to not decrease the site's performance much during the experiment. Evolution continuously creates improved candidates as it learns more about the system, whereas the Taguchi method generates a single set of candidates for the entire test. Evolution therefore provides continual improvement on the site even during the experiment. This principle is evident in the results of the third experiment, using the linear evaluator of equation 2 and the more complex genome of figure 9. As can be seen in figure 10, the Taguchi's candidates' average performance stays the same throughout the increasing traffic, whereas

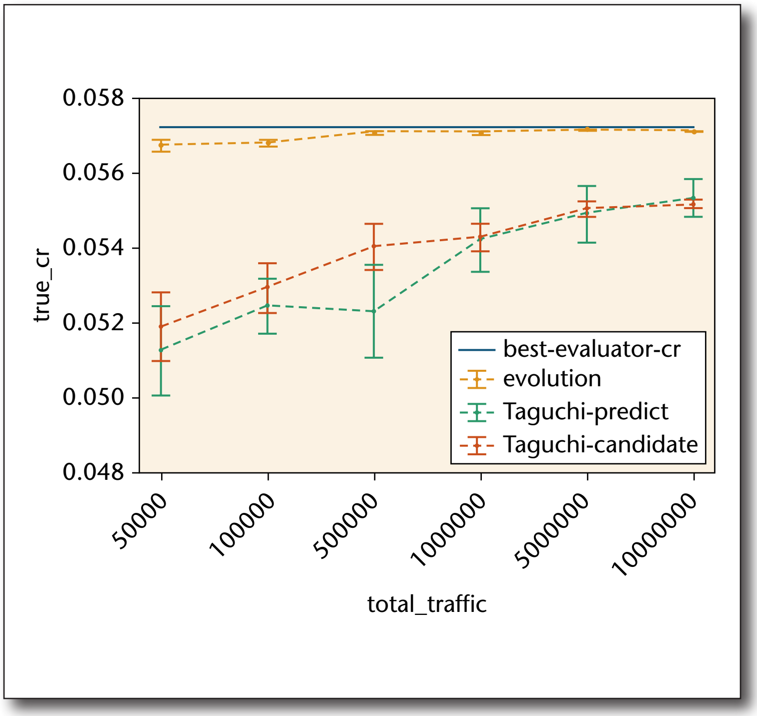


Figure 9. True Conversion Rate Performance of Evolution and the Taguchi Method with Increasing Amount of Traffic, with Interacting Variables.

The evolution line is the best candidate chosen by evolution algorithm; the Taguchi-predict line is the combined candidate from Taguchi variable analysis; and the Taguchi-candidate is the highest scored candidate in original input Taguchi array. Evolutionary optimization results in significantly better candidates at all traffic values. In addition, the Taguchi's predicted best candidates are similar to the best Taguchi candidate actually tested, suggesting that the interactions render Taguchi's construction process ineffective.

evolution's candidates perform, on average, better as the experiment progresses. It therefore forms a good approach in domains where performance matters during the experiment, in particular in campaigns that run only for a limited duration.

Traffic Allocation in Noisy Domains

When the Evolutionary CRO methods were taken out of the laboratory and into the real-world application, it became clear that there were new and interesting challenges that needed to be met. First, in the original Evolutionary CRO framework (Miikkulainen et al. 2017a; 2018), the evaluation of each candidate is performed in a static fashion. A fixed amount of traffic is allocated to each web design. This means even if a candidate is clearly bad based on a few visits, the system currently gives it the same amount of traffic as for good ones. A large amount of real traffic may be wasted by bad candidates, leading to more expensive evaluations. Second, during the normal evolutionary process, only weak statistical evidence is obtained. Therefore, there is a multiple hypotheses problem, that is, the winner candidate is most likely not the one with the best true conversion rate, but one that got lucky with evaluations. Third, the current evolutionary CRO technique is designed to identify a good candidate at the end of optimization. However, in some scenarios, like the limited-duration campaigns of figure 10, the goal for CRO is to make the overall conversion rate during optimization as high as possible. With uniform traffic allocation, bad candidates are tested as much as good ones, thereby reducing the overall conversion rate.

These issues can be addressed with a more intelligent traffic allocation based on the Multiarmed Bandit (MAB) approach. A general such approach, MAB-Evolutionary Algorithm (MAB-EA), will be developed in this section, as well as two specific methods, one for selecting the best candidate and another for maintaining high performance in campaign mode. The effectiveness of these methods will then be evaluated in simulation.[13] For more details, see Qiu and Miikkulainen (2019).

MAB Approach

The first goal is to develop a framework that allocates traffic dynamically in a more efficient way. MAB algorithms (Robbins 1952; Weber 1992; Auer, Cesa-Bianchi, and Fischer 2002; Bubeck, Munos, and Stoltz 2009; Audibert and Bubeck 2010) are well suited for this role. In MAB problem, a slot machine with multiple arms is given, and the gambler has to decide which arms to pull, how many times to pull each arm, and in which order to pull them, to maximize rewards. Each candidate web design can be regarded as an arm, and each visit to the website is equal to a pull. The reward of each visit to a single web design is assumed to follow an unknown but fixed Bernoulli distribution. The probability of getting reward 1 (the visited user is successfully converted) is p and the probability of getting reward 0 (the visited user is not converted) is $1 - p$, where p is the true conversion rate of that web design. Given a fixed budget of traffic (number of visits) for each generation, a Bernoulli MAB algorithm will then be invoked to allocate traffic to the current candidates.

The main effect of this method, MAB-EA, is that traffic is not wasted on bad candidates. A secondary effect is that it can instead be used to evaluate most promising candidates more accurately. The proposed framework is thus expected to both reduce the amount of traffic needed and improve overall optimization performance. Two specific instantiations of the framework will be described next, the first one for identifying a single best candidate at the end of evolution, and the second for maintaining high average performance during a campaign.

In the Best-Arm Identification (BAI) mode of MAB-EA, an additional BAI phase is applied after the evolution process has concluded. A MAB algorithm for pure exploration (successive rejects; Audibert and Bubeck 2010), will be performed on an elite archive, that is, the collection of top candidates over all generations. A single winner will be returned after the BAI phase. Although additional traffic is needed for running the BAI phase, this cost can be compensated by extracting a small portion of traffic from each previous generation (for example, ten percent).

In the Campaign mode, MAB-EA is extended with asynchronous statistics. Whereas measurements such as the total reward, average reward, number of pulls, and so forth of all the arms are usually initialized to zero, in Campaign mode all candidates that survive from the previous generation preserve these measurements and use them as the initial values in the current generation. Asynchronous MAB algorithm thus allocates more traffic to the existing elites without reevaluating them from scratch, focusing more on exploitation rather than exploration, and thus improving overall conversion rate.

Simulation Experiments

The simulator introduced in the Taguchi comparison section was used to evaluate the effectiveness of MAB-EA. The simulated website consisted of eight elements, with [5, 4, 2, 3, 4, 3, 3, 4] values. The control conversion rate is 0.05, and the effect of each element choice is within [–0.01, 0.01]. The mean conversion rate for all possible designs is 0.04997, and the maximum is 0.08494. Three MAB algorithms — Successive Rejects (SR), Thompson Sampling (TS), and Upper Confidence Bound 1 (UCB1) — were evaluated and compared with the standard uniform traffic allocation. The traffic budget for each generation is fixed at 10,000, the population size $K = 20$, mutation probability $C_m = 0.01$, and elite and parent percentages varied between 10 and 30 percent.

First, the main observation on the basic MAB-EA runs is that TS and SR increases both the best and the overall conversion rate compared with the standard method five to ten percent (the differences are statistically significant with $p < 0.05$ based on a t test on 500 independent runs). In contrast, because the average reward in the simulated CRO case is very low (for example, 0.05), UCB1 favors more exploration, which encourages evenly allocation of the traffic, thereby leading to similar performance as the standard method.

When evaluating the extension of MAB-EA to BAI, the basic MAB-EA methods have 11,000 visits per generation, BAI extensions have 10,000 visits per generation, and 10,000 additional visits in the BAI phase. The simulation is run for fifteen generations, which is typical for Ascend experiments where a best design needs to be found. As can be seen in figure 11, the BAI mode consistently improves over the Standard Method and the basic MAB-EA methods. It both converges faster early on, and explores more efficiently later. After Generation 10, BAI mode significantly outperforms MAB-EA even with less total traffic. BAI mode thus allows selecting a better winner, and estimates its future/true performance more accurately. It therefore provides an important improvement of the standard Ascend approach.

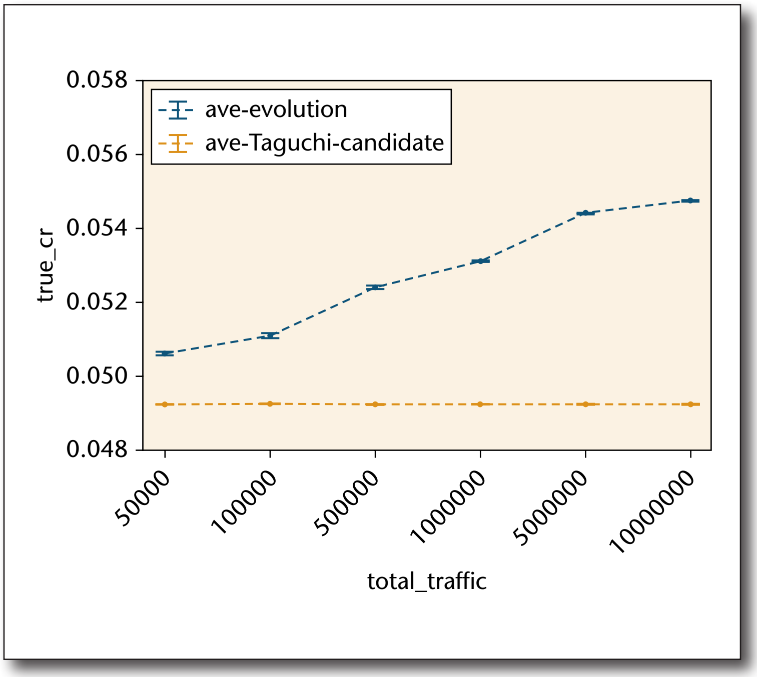


Figure 10. Average True Conversion Rate of Candidates with Evolution and Taguchi Methods during the Experiment.

While Taguchi candidates do not change, evolution continuously comes up with better candidates, thus increasing performance during the experiment. It therefore forms a good approach for campaigns with fixed duration as well.

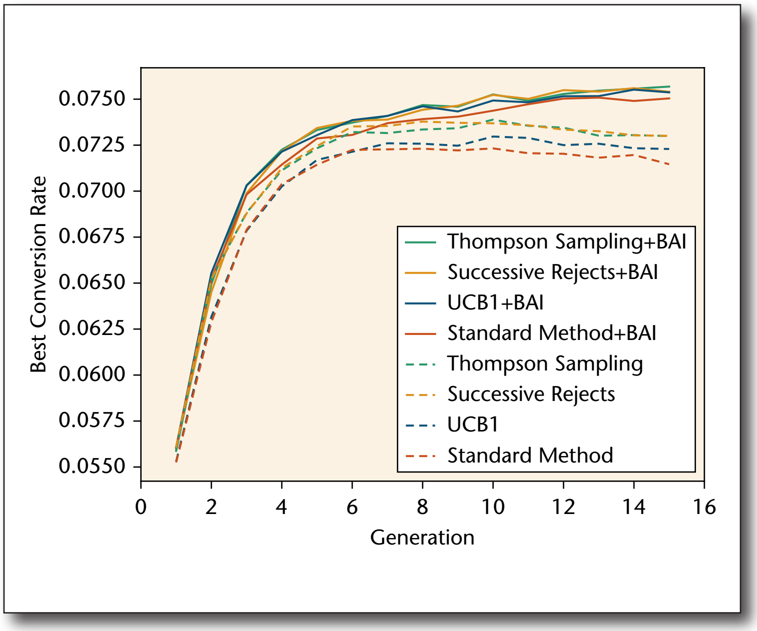


Figure 11. Best Conversion Rate Over Generations.

The methods with a BAI phase perform significantly better, that is, they allow identifying a candidate where true performance is significantly better than methods without a BAI phase. The results are averaged over 500 independent runs, and the performance differences between BAI variants and non-BAI variants are statistically significant with $p < 0.05$.

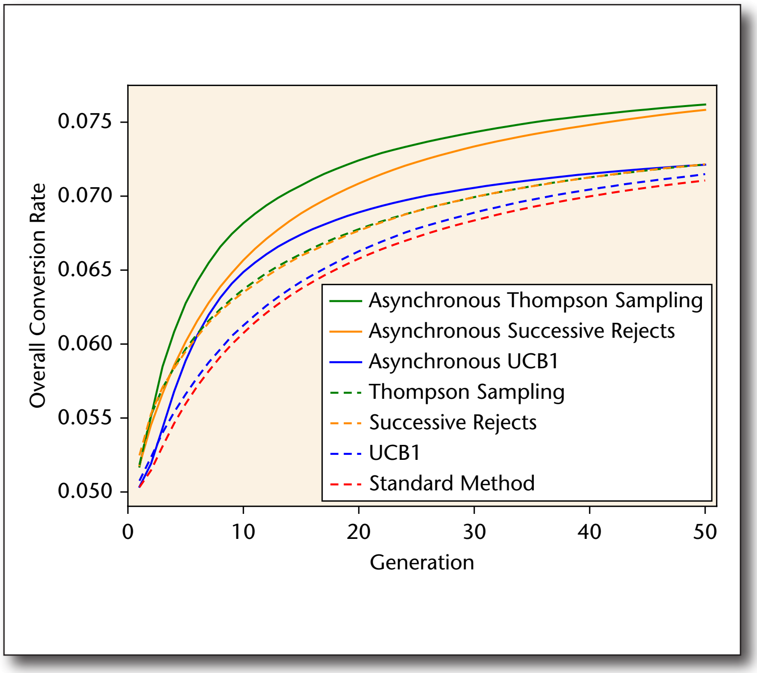


Figure 12. The Overall Conversion Rate for Entire Optimization Process in Campaign Mode.

The data point at generation *g* shows the overall conversion rate until generation *g*. The asynchronous versions of TS and SR perform significantly better than other variants, leading to better conversion rate over the entire campaign. The results are averaged over 500 independent runs, and the performance differences between asynchronous versions and original versions are statistically significant with $p < 0.05$ for all tested MAB algorithms.

In the Campaign mode experiments, SR, TS, and UCB1 are modified to run asynchronously and compared with their original versions, as well as with the Standard Method. Because Campaign mode usually runs for longer, the number of generations is set at fifty. As can be seen in figure 12, asynchronous SR and asynchronous TS perform significantly better than their original versions. For UCB1, the asynchronous version is better only in the early stages where exploration is more important.

These experiments therefore demonstrate how the MAB extension of Ascend can solve three general issues in evolutionary CRO — how to allocate the evaluation budget efficiently, how to select good final candidates reliably, and how to maintain high overall conversion rate during evolution.

Development, Deployment, and Maintenance

Ascend by Evolv is a software as a service application of evolutionary optimization. This section summarizes the Ascend team's experience in developing, deploying, and maintaining the software for the growing customer base.

The Ascend application is organized into three components: Runtime, which is the code deployed on a customers website to manipulate the page content and gather analytics data; Editor, which is the application that the customer uses to configure the Ascend experiment, specifying the pages to be tested and the types of changes to be made on them; and Evolution, the primary optimization module that decides what content to serve on the website.

Ascend was built and is maintained by a group of web developers, systems engineers, and data scientists. The team practices agile development methodologies as well as continuous deployment and integration. The team currently operates on a two-week sprint cycle, and splits backlog between the three primary components discussed above. The minimum viable product took six months to develop for a team of eight engineers (two front-end, three full-stack, two data-scientist, and one devops/pipeline engineer) and a project manager. The cost was roughly midlevel software engineering cost for the region (San Francisco Bay Area).

The main challenges in developing Ascend was to be able to render the changes on the webpages sufficiently fast, and minimize the central processing unit, bandwidth, and latency impact that this process causes on our customers websites. These difficulties were overcome with benchmarking tools, investments in latency-based routing systems, and through partnering with multiple high-performance content-delivery networks. In addition, implementation of evolutionary algorithms requires specialized knowledge in AI, and such talent is difficult to recruit and retain.

In terms of lessons learned, it turned out that every website and its rendering logic presents a new potential problem (and edge case) to solve. The team needed to develop a number of diagnostic tools to be able to respond to issues quickly, as opposed to a plan for mitigating all potential issues through defensive engineering. With web applications, issues will always arise, and the best plan is to prepare for issues and have a team on call to resolve them. In terms of methods, frequentist statistics requirements such as significance with $p < 0.05$ are not tenable in the highly variable environment of website traffic. Alternative methods of measuring statistical validity and selecting candidates are needed, such as the MAB methods described above, and a method based on averaging in the candidate neighborhoods (Miikkulainen et al. 2017b).

Ascend is maintained by a developer operations engineering team as well as software engineers that are responsible for each of the three components of the application. Updates are released roughly once every two weeks. The domain knowledge changes moderately over time. The data science needs to be updated to keep up with the growing customer base, and web analytics and browser support will require continual updates to keep up with the developments in these industries. The application is

Industry	No. of Values	No. of Elements	No. of Combinations	Length of Test	Conversion Rate Increase %
Annuities	11	3	48	12 weeks	24
Intimacy Apparel Retailer	15	4	160	8 weeks	38
Flower Retailer	16	8	256	8 weeks	35
Digital Commerce Payments	20	9	1,152	3 weeks	9
Web Search Results	26	10	10,368	6 weeks	22
Japanese Clothing Retailer	30	8	12,800	8 weeks	40
Classic Car Reseller	30	8	28,800	3 weeks	434
Entertainment Ecommerce	32	8	77,760	5 weeks	50
Comparison Shopping	30	8	241,920	9 weeks	31
Leading Mobile Network	42	9	1,296,600	6 weeks	75
Australian Beauty Retailer	48	13	1,382,400	8 weeks	45

During its first year as a commercial product, Ascend has been used to optimize a diverse set of web interfaces consistently and significantly, with typical CR gains of 20 to 50 percent, and sometimes over 400 percent.

Table 2. Examples of Ascend Applications Across Industries and Search Space Sizes.

modularized so that releases can be pushed to components of the application without interacting with the critical path where not needed. For example, evolution is built as a service and therefore can be updated without impacting the rest of the application. Changes to evolution methods can be tested in simulation based on historical data before deploying them in the application itself.

Discussion and Future Work

During its first year, Ascend was applied to numerous web interfaces across industries and search-space sizes. The results were remarkably reliable. In all cases the conversion rates were improved significantly over control, in some cases over fourfold (table 2). Although Ascend was expected to excel in search spaces with millions of combinations, somewhat surprisingly it also finds improvements even in spaces with a few dozen combinations — suggesting that human intuition in this domain is limited, and automated methods can help.

The main challenge is indeed the human element, in two ways. First, web designers, who are used to A/B and multivariate testing, often try to minimize the search space as much as possible, that is, limit the number of elements and values, thereby not giving evolution much space to explore and discover the most powerful solutions. Second, because it often takes only a couple of generations for evolution to discover significant improvement, the designers are likely to terminate evolution early, instead of letting it optimize the designs fully. Utilizing evolutionary search as a tool requires a different kind of thinking; as designers become more familiar with it, we believe they will be able to take advantage of the full power of evolutionary search, reaching more refined results.

Currently Ascend delivers one best design, or a small number of good ones, in the end as the result, again in keeping with the A/B testing tradition. In many cases there are seasonal variations and other long-term changing trends, making the performance of good designs gradually decay. It is possible to counter this problem by running the optimization again every few months. However, a new paradigm of "always-on" would be more appropriate. Evolutionary optimization can be run continuously at a low volume, keeping up with changing trends (that is, through dynamic evolutionary optimization; Branke 2002). New designs can then be adopted periodically when their performance exceeds old designs significantly.

Also, in some cases the customer wants to run a limited campaign, driving traffic to the site for example, for a few weeks, after which time the web interface will no longer be needed. Instead of optimizing the final web interface design, conversions need to be optimized over all designs tested during evolution. As seen in figure 7, the average performance of all candidates tested usually arises above the control very quickly, and Ascend can therefore already be used for campaigns as is. However, knowing that every candidate counts toward performance, traffic can be allocated more efficiently, to optimize campaign performance instead of future performance. The MAB methods described are a promising approach to that end.

Furthermore, currently Ascend optimizes a single design to be used with all future users of a mobile or desktop site. An interesting extension would be to take user segmentation (Yankelovich and Meer 2006) into account, and evolve different pages for different kinds of users. Moreover, such a mapping from user characterizations to page designs can be automatic. A mapping system such as a neural network can take user variables such as location, time, device, and any past history with the site as inputs, and generate the vector of elements and their values as outputs. Neuroevolution (Floreano, Dürr, and Mattiussi 2008; Lehman and Miikkulainen 2013b) can discover optimal such mappings, in effect evolve to discover a dynamic, continuous segmentation of the user space. Users will be shown designs that are likely to convert well based on experience with other users with similar characteristics, continuously and automatically. It will be possible to analyze such evolved neural networks and discover what variables are most predictive, characterize the main user segments, and thereby develop an in-depth understanding of the opportunity.

Finally, the Ascend approach is not limited to optimizing conversions. Any outcome that can be measured, such as revenue or user retention, can be optimized. The approach can also be used in a different role, such as optimizing the amount of resources spent on attracting users, such as ad placement and selection, adword bidding, and e-mail marketing. The approach can be seen as a fundamental step in bringing machine optimization into e-commerce, and demonstrating the value of evolutionary computation in real-world problems.

Conclusion

Ascend by Evolv is the first automated system for massively multivariate conversion optimization — replacing A/B with AI. Ascend scales up interactive evolution by testing a large number of candidates in parallel on real users. Human designers specify the search space, and evolutionary optimization finds effective designs in that space, including design principles that humans tend to overlook, and interactions that current multivariate methods miss. Ascend has been applied to numerous web interfaces across industries and search space sizes and has been able to improve them consistently and significantly. In the future, it should be possible to extend it to continuous optimization, limited-time campaigns, and user segmentation as well.

Notes

1. Ascend by Evol.: Evolv Solution [accessed 2 January 2020], evolv.ai/solution.

2. Adobe: Multivariate Test [accessed 2 January 2020], docs.adobe.com/content/help/en/target/using/activities/multivariatetest/multivariate-testing.html.

3. Sentient: It's Not A/B, It's AI [2 October 2016 version accessed 2 January 2020], web.archive.org/web/20161001172820/http://www.sentient.ai/ascend.

4. Builtwith: A/B Testing Usage [accessed 2 January 2020], trends.builtwith.com/analytics/a-b-testing.

5. Optimizely: Out-experiment. Outperform. [accessed 2 January 2020], www.optimizely.com.

6. VWO: Accelerate Growth by Delivering Optimized Digital Experiences [accessed 2 January 2020], vwo.com.

7. Mixpanel: Grow your business by learning why users convert, engage, and retain [accessed 2 January 2020], mixpanel.com.

8. Google: What Is 'Probability to Beat Baseline'? [accessed 2 January 2020], support.google.com/optimize/answer/7405044.

9. SplitMetrics: Chance to Beat Control [accessed 2 January 2020], splitmetrics.com/resources.

10. VWO: How Does VWO Calculate the Chance to Beat [accessed 2 January 2020], vwo.com/knowledge/how-does-Vwo-calculate-the-chance-to-beat.

11. For an animated demo of this experiment, see evolution.ml/demos/ascend [accessed 2 January 2020].

12. Kufeld, W.F.: Orthogonal Arrays [accessed 2 January 2020], support.sas.com/techsup/technote/ts723.html.

13. For animated demos of this process, see evolution.ml/demos/mab1,evolution.ml/demos/mab2,evolution.ml/demos/bai [accessed 2 January 2020].

References

Ash, T.; Page, R.; and Ginty, M. 2012. *Landing Page Optimization: The Definitive Guide to Testing and Tuning for Conversions*. Second edition. Hoboken, NJ: Wiley.

Audibert, J.-Y., and Bubeck, S. 2010. Best Arm Identification in Multi-Armed Bandits. *In* Proceedings of the 23rd Annual Conference on Learning Theory (COLT). 41–53.

Auer, P.; Cesa-Bianchi, N.; and Fischer, P. 2002. Finite-Time Analysis of the Multiarmed Bandit Problem. *Machine Learning* 47(2/3): 235–56. doi.org/10.1023/A:1013689704352.

Brabham, D. C. 2013. *Crowdsourcing*. Cambridge, MA: MIT Press. doi.org/10.7551/mitpress/9693.001.0001.

Branke, J. 2002. *Evolutionary Optimization in Dynamic Environments*. Berlin: Springer. doi.org/10.1007/978-1-4615-0911-0.

Brouwer, A. E.; Cohen, A. M.; and Nguyen, M. V. 2006. Orthogonal Arrays of Strength 3 and Small Run Sizes. *Journal of Statistical Planning and Inference* 136(9): 3268–80. doi.org/10.1016/j.jspi.2004.12.012.

Bubeck, S.; Munos, R.; and Stoltz, G. 2009. Pure Exploration in Multi-Armed Bandits Problems. In *Proceedings, Algorithmic Learning Theory: 20th International Conference*. Gavaldá, R.; Lugosi, G.; Zeugmann, T.; and Zilles, S., Eds. Berlin, Heidelberg: Springer. 23–37.

eMarketer. 2016. US Digital Ad Spending to Surpass TV This Year. emarketer September 13. www.emarketer.com/Article/US-Digital-Ad-Spending-Surpass-TV-this-Year/1014469.

Floreano, D.; Dürr, P.; and Mattiussi, C. 2008. Neuroevolution: From Architectures to Learning. *Evolutionary Intelligence* 1(1): 47–62. www.researchgate.net/publication/37452617_Neuroevolution_From_architectures_to_learning.

Hedayat, A. S.; Sloane, N. J. A.; and Stufken, J. 2018. *Orthogonal Arrays: Theory and Applications*. Berlin: Springer Science and Business Media. www.springer.com/gp/book/9780387987668.

Hodjat, B., and Shahrzad, H. 2013. Introducing an Age-Varying Fitness Estimation Function. In *Genetic Programming Theory and Practice X*. Riolo, R.; Vladislavleva, E.; Ritchie, M. D.; and Moore, J. H., Eds. Genetic and Evolutionary Computation Book Series. New York, NY: Springer. 59–71. doi.org/10.1007/978-1-4614-6846-2_5.

Jiang, J.; Legrand, D.; Severn, R.; and Miikkulainen, R. 2018. *A Comparison of the Taguchi Method and Evolutionary Optimization in Multivariate Testing*. ArXiv E-Prints. arxiv.org/abs/1808.08347.

Kohavi, R., and Longbotham, R. 2016. Online Controlled Experiments and A/B Tests. In *Encyclopedia of Machine Learning and Data Mining*. Sammut, C., and Webb, G.I., Eds. New York, NY: Springer. exp-platform.com/encyclopediamldm/.

Kohavi, R., and Thomke, S. 2017. The Surprising Power of Online Experiments. *Harvard Business Review* 95(5): 74–82. hbr.org/2017/09/the-surprising-power-of-online-experiments.

Lehman, J., and Miikkulainen, R. 2013a. Boosting Interactive Evolution Using Human Computation Markets. In *Proceedings of the 2nd International Conference on the Theory and Practice of Natural Computation*. Berlin: Springer. doi.org/10.1007/978-3-642-45008-2_1.

Lehman, J., and Miikkulainen, R. 2013b. Neuroevolution. *Scholarpedia* 8(6): 30977. doi.org/10.4249/scholarpedia.30977.

Miikkulainen, R.; Iscoe, N.; Shagrin, A.; Cordell, R.; Nazari, S.; Schoolland, C.; Brundage, M.; Epstein, J.; Dean, R.; and Lamba, G. 2017a. Conversion Rate Optimization Through Evolutionary Computation. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2017)*. New York, NY: Association for Computing Machinery. doi.org/10.1145/3071178.3071312.

Miikkulainen, R.; Iscoe, N.; Shagrin, A.; Rapp, R.; Nazari, S.; McGrath, P.; Schoolland, C.; Achkar, E.; Brundage, M.; Miller, J., et al. 2018. Sentient Ascend: AI-Based Massively Multivariate Conversion Rate Optimization. In *Proceedings of the Thirtieth Innovative Applications of Artificial Intelligence Conference*. Menlo Park, CA: Association for the Advancement of Artificial Intelligence.

Miikkulainen, R.; Shahrzad, H.; Duffy, N.; and Long, P. 2017b. How to Select a Winner in Evolutionary Optimization? In *Proceedings of the Institute of Electrical and Electronics Engineers (IEEE) Symposium Series in Computational Intelligence*. New York, NY: Institute of Electrical and Electronics Engineers. doi.org/10.1109/SSCI.2017.8280904.

Qiu, X., and Miikkulainen, R. 2019. Enhancing Evolutionary Conversion Rate Optimization via Multi-Armed Bandit Algorithms. In *Proceedings of the 31st Conference on Innovative Applications of Artificial Intelligence*. Menlo Park, CA: Association for the Advancement of Artificial Intelligence. doi.org/10.1609/aaai.v33i01.33019581.

Rao, R. S.; Kumar, C. G.; Prakasham, R. S.; and Hobbs, P. J. 2008. The Taguchi Methodology as a Statistical Tool for Biotechnological Applications: A Critical Appraisal. *Biotechnology Journal* 3(4): 510–23. doi.org/10.1002/biot.200700201.

Robbins, H. 1952. Some Aspects of the Sequential Design of Experiments. *Bulletin of the American Mathematical Society* 58(5): 527–35. doi.org/10.1090/S0002-9904-1952-09620-8.

Salehd, K., and Shukairy, A. 2011. *Conversion Optimization: The Art and Science of Converting Prospects to Customers*. Sebastopol, CA: O'Reilly Media. shop.oreilly.com/product/0636920000167.do.

Secretan, J.; Beato, N.; D'Ambrosio, D. B.; Rodriguez, A.; Campbell, A.; Folsom-Kovarik, J. T.; and Stanley, K. O. 2011. Picbreeder: A Case Study in Collaborative Evolutionary Exploration of Design Space. *Evolutionary Computation* 19(3): 373–403. doi.org/10.1162/EVCO_a_00030.

Shahrzad, H.; Hodjat, B.; and Miikkulainen, R. 2016. Estimating the Advantage of Age-Layering in Evolutionary Algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2016)*. New York, NY: Association for the Advancement of Artificial Intelligence. doi.org/10.1145/2908812.2908911.

Takagi, H. 2001. Interactive Evolutionary Computation: Fusion of the Capacities of EC Optimization and Human Evaluation. *Proceedings of the Institute of Electrical and Electronics Engineers* 89(9): 1275–96. doi.org/10.1109/5.949485.

Weber, R. 1992. On the Gittins Index for Multiarmed Bandits. *Annals of Applied Probability* 2(4): 1024–33. doi.org/10.1214/aoap/1177005588.

Yankelovich, D., and Meer, D. 2006. Rediscovering Market Segmentation. *Harvard Business Review* 84(2): 122–31. hbr.org/2006/02/rediscovering-market-segmentation.

Risto Miikkulainen is a Professor of Computer Science at the University of Texas at Austin and Associate Vice President of evolutionary AI at Cognizant. His current research focuses on methods and applications of neuroevolution, as well as neural network models of natural language processing and vision. At Cognizant, and previously as Chief Technology Officer of Sentient Technologies, he is scaling up these approaches to real-world problems. He received an MS in Engineering from the Helsinki University of Technology (now Aalto University) in 1986, and a PhD in Computer Science from the University of California-Los Angeles in 1990.

Jonathan Epstein is Chief Strategy Officer for Evolv Technologies. Previously Chief Marketing Officer and Senior Vice President (International) at Sentient, he was intimately involved with the development and launch of Ascend. Before working at Evolv and Sentient, Epstein has held key executive positions at the intersection of technology and media, including President of Omek Interactive, Chief Executive Officer of GameSpot, Chief Executive Officer of Double Fusion, and Senior Vice President of IGN Entertainment. He has authored multiple patents in fields ranging from gesture control to in-game advertising to remotely operated underwater vehicles. Epstein graduated from Harvard University with an AB in Physical Sciences.

Babak Hodjat is Vice President of Evolutionary AI at Cognizant, and former cofounder and Chief Executive Officer of Sentient and a cofounder of Sentient Investment Management. He is a serial entrepreneur, having started a number of Silicon Valley companies as main inventor and technologist. Before cofounding Sentient, Hodjat was Senior Director of Engineering at Sybase iAnywhere, where he led mobile solutions engineering, and a cofounder, Chief Technology Officer, and board member of Dejima Inc. Hodjat is the primary inventor of Dejima's agent-oriented technology applied to intelligent interfaces for mobile and enterprise computing — the technology behind Apple's Siri. He has publications and patents in numerous fields of AI, including natural language processing, machine learning, genetic algorithms, and distributed AI. He holds a PhD in Machine Intelligence from Kyushu University, Fukuoka, Japan.

Neil Iscoe was the Chief Executive Officer and cofounder of Digital Certainty, the company that created the original version of Ascend. After the product was sold to Sentient Technologies, he became the product's general manager. Previously, he was the Associate Vice President of Research and Director of Technology Commercialization at the University of Texas, where he was responsible for creating commercialization entities and marketable products from university research. In 2011, he left the university to build Ascend. He has an MS and PhD in Computer Science, with a Specialization in Systems and AI from the University of Texas.

Jingbo Jiang was a intern in Sentient Technologies during the summer of 2018, working on an evolution algorithm and its application to web page design, and in particular the comparisons with the Taguchi method. Her background is in machine learning, computer vision, and language processing.

She earned her MS in Data Science from the University of Pennsylvania in 2019 and her BE in Electrical Engineering from Beihang University, China in 2017.

Diego Legrand was a Machine Learning Engineer at Sentient Technologies. His work on Bayesian models changed Ascend's way of finding top performing candidates and significantly increased the learning speed of the system. Diego has a masters in Applied Mathematics from Centrale University in Paris, France.

Sam Nazari is the Vice President of Customer Success at Evolv.AI. He leads the global team that works closely with clients to help them understand and integrate AI across their enterprises (from large Fortune-500 companies to medium-sized businesses spanning multiple verticals). This ranges from explaining best-use cases for AI in marketing through to how to prepare their data, design and implement the technology, and seek compliance with their Information Technology teams, to the successful rollout of AI to help drive revenue. Nazari has a BS in Computer and Electrical Engineering from the University of Utah.

Xin Qiu is a senior research scientist at Cognizant and previously at Sentient. His research interests include evolutionary computation, probabilistic modeling, bandit algorithms, and reinforcement learning. He earned his PhD from the National University of Singapore in 2016 and his BE from Nanjing University in 2012.

Michael Scharff is the chief executive officer for Evolv Technologies, the AI firm behind the Ascend autonomous website optimization platform. Scharff brings over two decades of digital commerce and retail experience, with leadership roles at some of the most well-known retailers in North America including Sears Canada, Toys R Us, Staples, and Best Buy. He has a wealth of experience in all aspects of retailing and across numerous industry verticals and channels. Scharff has built and managed highly successful omni-channel and global eCommerce businesses, and led teams in merchandising, digital marketing, innovation, and other functional areas.

Cory Schoolland has over a decade of experience heading marketing design efforts for San Francisco-based software-as-a-service companies including RichRelevance, Sentient Technologies, and Evolv Technologies. He believes in the power of good design to improve our lives, and enjoys combining words, shapes, images, and colors to tell a story, or taking existing content and making it beautiful.

Rob Severn is the director of product at the Evolv. He is responsible for understanding the optimization market's problems and needs to better guide Evolv's optimization product. Severn received a BA in Mathematics from Cambridge University in 2006.

Aaron Shagrin has been working with technology companies, large and small, for over 20 years. He has been a part of multiple startups, Fortune-500 companies, and private equity firms. He has a deep background in product management and strategy, acquisitions, alliances, and business development and sales. He has a BBA from The University of Texas.