Sentient Ascend: AI-Based Massively Multivariate Conversion Rate Optimization

Risto Miikkulainen,[1,2] **Neil Iscoe,**[1] **Aaron Shagrin,**[1] **Ryan Rapp,**[1]
Sam Nazari,[1] **Patrick McGrath,**[1] **Cory Schoolland,**[1] **Elyas Achkar,**[1]
Myles Brundage,[1] **Jeremy Miller,**[1] **Jonathan Epstein,**[1] **Gurmeet Lamba**[1]
[1]Sentient Technologies, Inc.
[2]The University of Texas at Austin

Abstract

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, or evaluating all combinations of two or three variables through multivariate testing. Traditional CRO is thus limited to a small fraction of the design space only. This paper describes Sentient Ascend, 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 on line with real users, making it possible to discover and utilize 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 four-fold improvements over human design. Ascend can therefore be seen as massively multivariate CRO made possible by AI.

Introduction

In e-commerce, designing web interfaces (i.e. web pages and interactions) that convert as many users as possible from casual browsers to paying customers is an important goal (Ash, Page, and Ginty 2012; Salehd and Shukairy 2011). 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, i.e. 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 (e.g. 2-3); the rest of the design space remains unexplored.

This paper describes an AI-assisted technology for conversion optimization based on evolutionary computation. This technology is implemented in Ascend, a conversion optimization product by Sentient Technologies, deployed in numerous e-commerce websites of paying customers since September 2016 (Sentient Technologies 2017). 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 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. 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 20-200% over human design are routine using this approach (Sentient Technologies 2017). These results are reliable across industries and search-space sizes.

This paper describes the technology underlying Ascend, presents an example use case, 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, i.e. the study of how web interfaces should be designed so that they are as effective as possible in converting users from ca-

sual 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 link in an email, website, or desktop, mobile, or social media application (Ash, Page, and Ginty 2012; Salehd and Shukairy 2011). 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 2-4% (i.e. 2-4% of the users that come to the site convert within 30 days). In 2014, only 18% of the top 10,000 e-commerce sites did any conversion optimization; in January 2017, 30% of them did so (Builtwith 2017). The growth is largely due to available conversion optimization tools, such as Optimizely, Visual Website Optimizer, Mixpanel, and Adobe Target (Builtwith 2017). 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, while 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, since each step in the process requires statistical significance, only a few designs can be tested. Third, each improvement step amounts to one step in hillclimbing; such a process can get stuck in local maxima. Fourth, the process is aimed at reducing false positives and therefore increases false negatives, i.e. designs with good ideas may be overlooked. Fifth, while the tools provide support for multivariate testing, in practice only a few combinations can be tested (e.g. five possible values for two elements, or three possible values for three elements). As a result, it is difficult to discover and utilize 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; 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.

Figure 1: Elements and Values of an Example Web Page Design. In this example, 13 elements each have 2-4 possible values, resulting in 1.1M combinations.

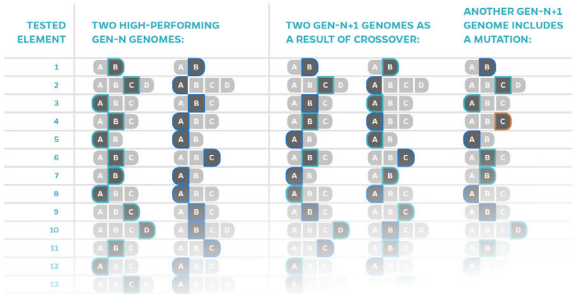

Figure 2: Genetic Encoding and Operations on Web Interface Candidates. The pages are represented as concatenations of their element values with one-hot encoding. Crossover and mutation operate on these vectors as usual, creating new combinations of values.

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 (i.e. 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, 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 Figure 1, logo size, header image, button color, content order are such elements, and they can each take on 2-4 values.

Ascend searches for good designs in the space of possible combinations of these values. This space is combinatorial, and can be very large, e.g. 1.1M in this example. Interestingly, it is exactly this combinatorial nature that makes webpage optimization a good application for evolution: Even though human designers have insight into what values to use, their combinations are difficult to predict, and need to be discovered by 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 to 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.

Evolutionary Process

Each page is represented as a genome, as shown for two example pages in Figure 2 (left side). The usual genetic operations of crossover (re-combination of the elements in the two genomes; middle) and mutation (randomly changing one element in the offspring; right side) 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, i.e. whether it should serve as a parent in the next generation, or should be discarded. A process similar to age-layering (Shahrzad, Hodjat, and Miikkulainen 2016; Hodjat and Shahrzad 2013) 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 2000. 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, i.e. thousands of pages instead of millions or billions.

Online Evolution

While 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 3 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 show to which user, so that they get to see the same design if they return within a certain time limit (e.g. 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.

The initial design for this page is shown in the left side of Figure 4. 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%, 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 5). While much larger search spaces are possible, this example represents a mid-size space common with many current sites.

The initial population of 37 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 60 days, or four generations, altogether testing 111 candidates

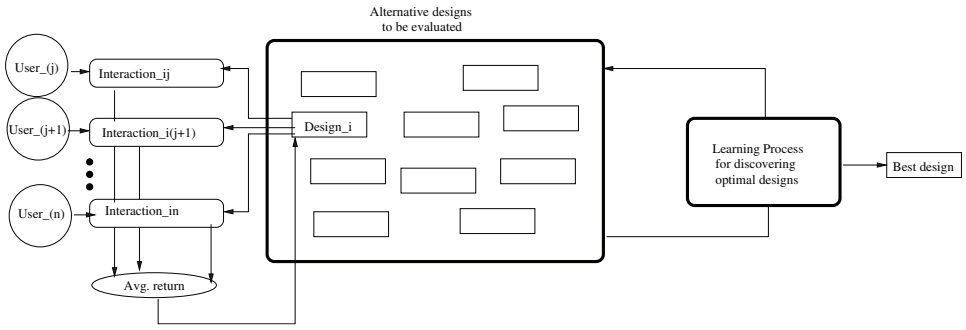

Figure 3: Overall Architecture of the Online Evolution System. The outcome of each interaction (i.e. whether the user converted or not) constitutes one evaluation of a design. Many such evaluations ij are run in parallel with different users j and averaged to estimate how good the design i is. After 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.

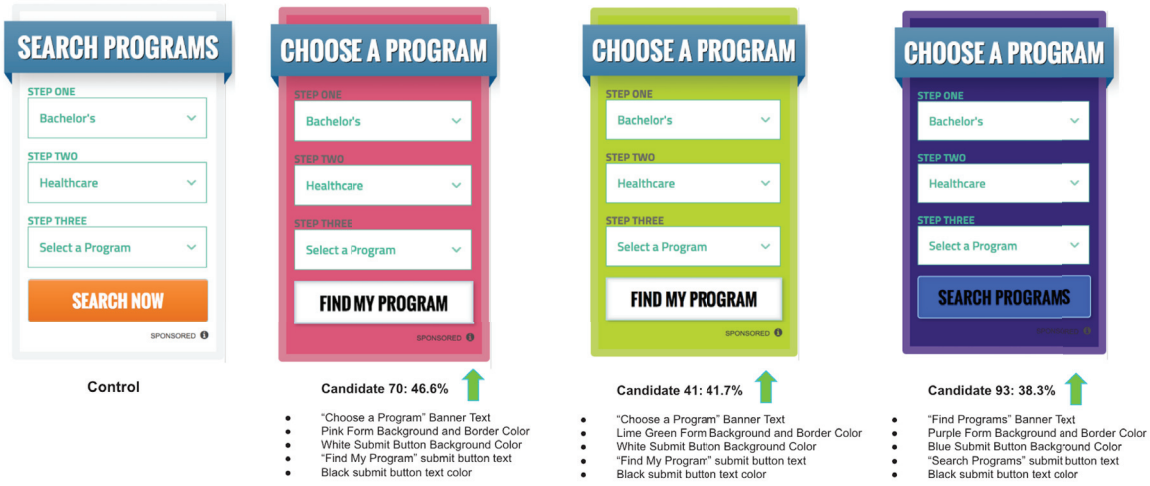

Figure 4: The control design and three best evolved designs. After 60 days of evolution with 599,008 user interactions, a design for the search widget was found that converted 46.6% better than the control (5.61% vs. 8.22%), as well as other good designs. Much of the improvement was based on discovering a combination of colors that draws attention to the widget and makes the call to action clear.

with 599,008 user interactions total. The estimated conversion rates of the candidates over this time are shown in Figure 6. This figure demonstrates that evolution was successful in discovering significantly better candidates than control.

As an independent verification, the three top candidates in Figure 4 were then subjected to an A/B test using Optimizely. In about 6500 user interactions, the best candidate was confirmed to increase the conversion rate by 43.5% with greater than 99% significance (and the other two by 37.1% and 28.2%)—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 utilize 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 (i.e. "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 (e.g. 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

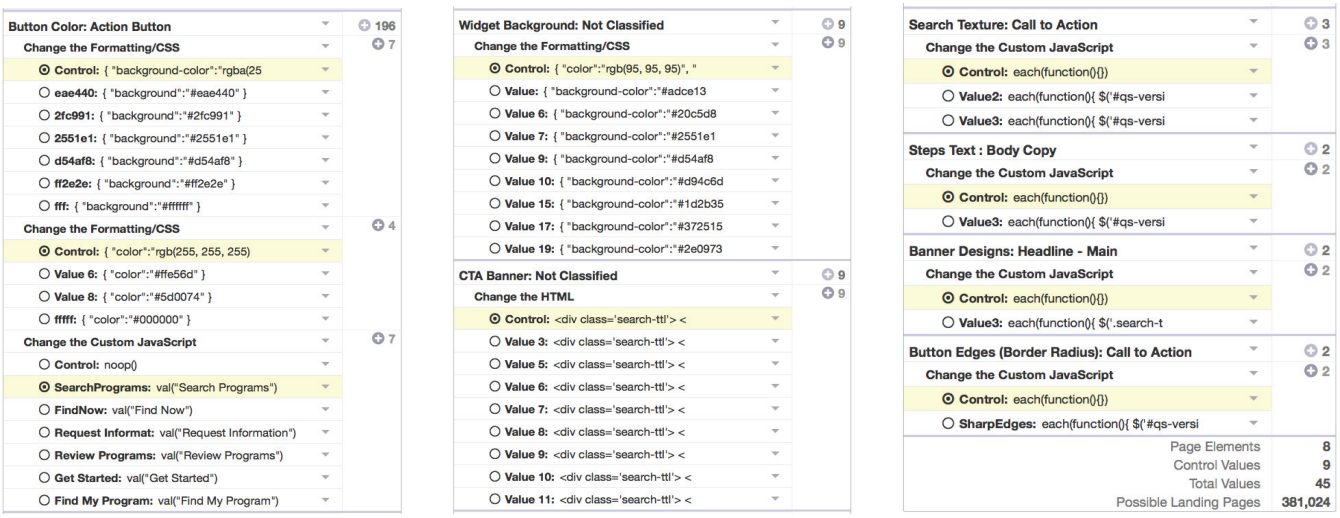

Figure 5: A screenshot of the user interface for designing Ascend experiments, showing the elements and values in the education program case study. Nine elements with two to nine different values each result in 381,024 potential web page designs; the first value in each element is designated as the control. This is a mid-size problem typical of current web interface designs.

typical human designs. Remarkably, in doing so Ascend succeeded in creating a sense of urgency that is missing from the control design (Figure 7), suggesting that Ascend can discover effective design principles of its own.

Development, Deployment, and Maintenance

Sentient Ascend is a software as a service (SaaS) 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: (1) Runtime: The code deployed on a customers website to manipulate the page content and gather analytics data. (2) Editor: 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. (3) 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 mid-level SWE 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 CPU, 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 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 (Miikkulainen et al. 2017).

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 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.

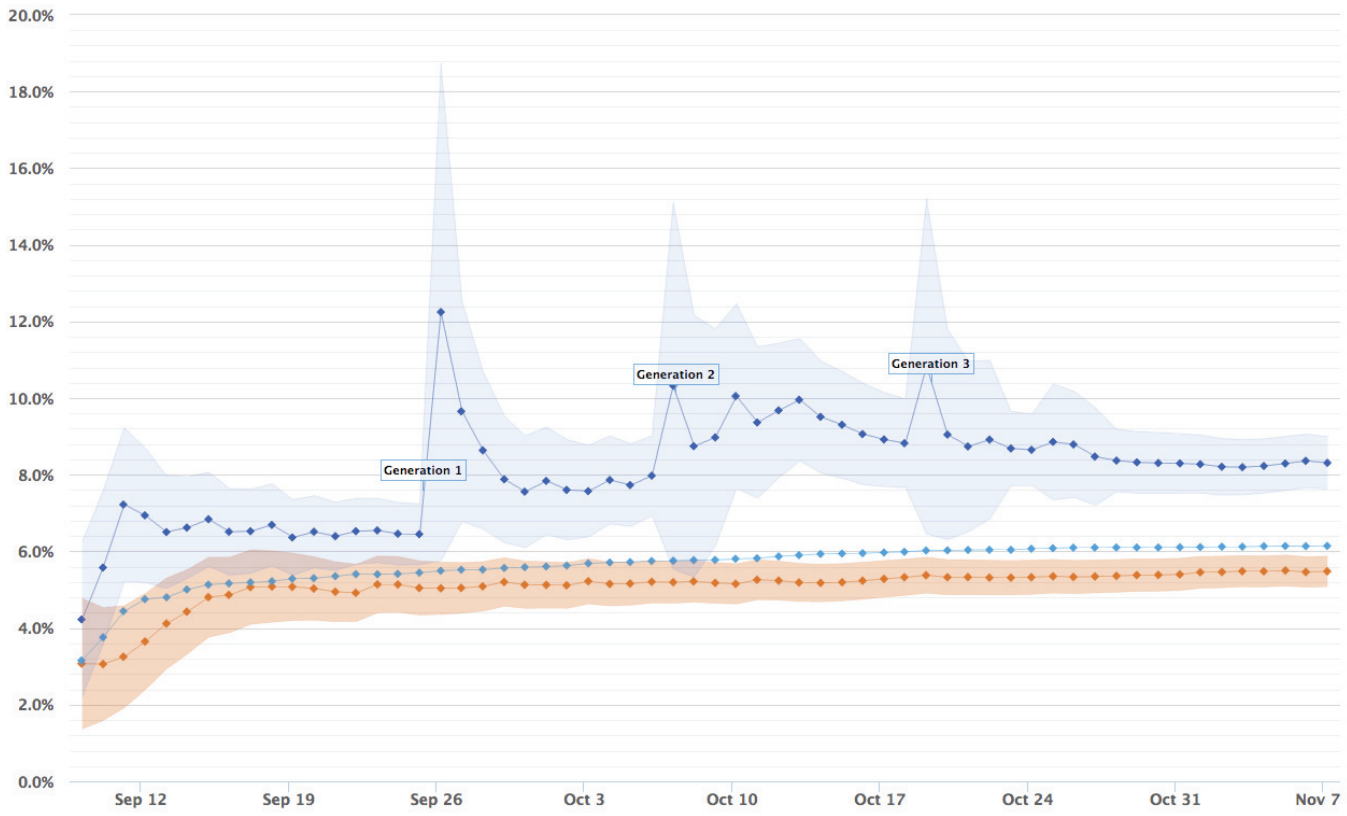

Figure 6: Estimated Conversion Rates through the 60-day Online Evolution Run. Days are in the x-axis and the conversion rate on the y axis. The dark blue dots (on top) indicate the current best candidate, the light blue dots (in the middle) an average of all currently active candidates, and the orange dots (at the bottom) the estimated performance of the control design. The shaded areas display the 95% confidence intervals (from the binomial distribution with the observed mean). The dark 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 2000 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 towards 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.

.

Discussion and Future Work

During its first year, Ascend has been applied to numerous web interfaces across industries and search-space sizes. The results have been remarkably reliable: In all cases the conversion rates were improved significantly over control, in some cases over four-fold (Table 1). 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, i.e. 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

Figure 7: 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% 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.

Industry	# of values	# of elements	# of combinations	Length of test	CR 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

Table 1: Examples of Ascend applications across industries and search space sizes. 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 20-50%, and sometimes over 400%.

"always-on" would be more appropriate: Evolutionary optimization can be run continuously at a low volume, keeping up with changing trends (i.e. 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 e.g. 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 6, 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, in order to optimize campaign performance instead of future performance.

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, any past history with the site as inputs, and generate the vector of elements and their values as outputs. Neuroevolution (Lehman and Miikkulainen 2013b; Floreano, Dürr, and Mattiussi 2008) 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 email 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

Sentient Ascend 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. 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.

References

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

Brabham, D. C. 2013. *Crowdsourcing*. Cambridge, MA: MIT Press.

Branke, J. 2002. *Evolutionary Optimization in Dynamic Environments*. Berlin: Springer.

Builtwith. 2017. A/B testing usage. Retrieved 1/9/2017.

eMarketer. 2016. Us digital ad spending to surpass tv this year. Retrieved 2/1/2017.

Floreano, D.; Dürr, P.; and Mattiussi, C. 2008. Neuroevolution: From architectures to learning. *Evolutionary Intelligence* 1:47–62.

Hodjat, B., and Shahrzad, H. 2013. Introducing an age-varying fitness estimation function. In Riolo, R.; Vladislavleva, E.; Ritchie, M. D.; and Moore, J. H., eds., *Genetic Programming Theory and Practice X*. New York: Springer. 59–71.

Kohavi, R., and Longbotham, R. 2016. Online controlled experiments and A/B tests. In Sammut, C., and Webb, G. I., eds., *Encyclopedia of Machine Learning and Data Mining*. New York: Springer.

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.

Lehman, J., and Miikkulainen, R. 2013b. Neuroevolution. *Scholarpedia* 8(6):30977.

Miikkulainen, R.; Shahrzad, H.; Duffy, N.; and Long, P. 2017. How to select a winner in evolutionary optimization? In *Proceedings of the IEEE Symposium Series in Computational Intelligence*. IEEE.

Salehd, K., and Shukairy, A. 2011. *Conversion Optimization: The Art and Science of Converting Prospects to Customers*. Sebastopol, CA: O'Reilly Media, Inc.

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* 19:345–371.

Sentient Technologies. 2017. It's not A/B, i's AI. Retrieved 1/9/2017.

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: ACM.

Takagi, H. 2001. Interactive evolutionary computation: Fusion of the capacities of EC optimization and human evaluation. *Proceedings of the IEEE* 89(9):1275–1296.

Yankelovich, D., and Meer, D. 2006. Rediscovering market segmentation. *Harvard Business Review* 84(2).