



A multi-objective evolutionary approach towards automated online controlled experiments[☆]

Jie J.W. Wu^{*}, Thomas A. Mazzuchi, Shahram Sarkani

Department of Engineering Management and Systems Engineering at the George Washington University, Washington, DC 20052, United States of America

ARTICLE INFO

Article history:

Received 16 October 2022
Received in revised form 19 March 2023
Accepted 8 April 2023
Available online 20 April 2023

Keywords:

Multiple objective evolutionary algorithm
A/B testing
Online controlled experiments

ABSTRACT

Due to the complexity of the web and mobile software applications, engineers rely heavily on A/B testing (i.e., online controlled experiments) to evaluate and measure the impact of the new changes. However, creating and evaluating A/B tests is a time-consuming, error-prone, and costly manual activity. For the problem of automating A/B testing, there is no solution in literature that evaluates A/B testing results as multi-criteria instead of single criteria. The current single criteria based methods overly simplify the A/B test results, as in the web/mobile application industry, a typical A/B test in practice has multiple metrics instead of one metric. In this paper, we describe the Variant Creation and Evaluation (VCE) problem in A/B testing and propose MOVSW (Multi-Objective Variant Selection Wrapper) that utilizes Multi-Objective Evolutionary Algorithm (MOEA) to automate the process of creating and selecting variants as launch candidates in A/B testing. Given a set of parameters we would like to optimize, the control group in the A/B test uses the existing values of these parameters in the system. The MOVSW automatically creates variants of different parameter values and conducts A/B testing to evaluate these variants against the control group with existing parameter values based on multiple measurements. The outputs of the MOVSW are a list of non-dominated candidate variants (also known as Pareto optimal set) that lead to potential improvements on key measures according to the A/B test results. These candidate variants could be then reviewed by stakeholders to determine the final variant to be launched. We designed and conducted a case study to comprehensively evaluate the effectiveness of five MOEAs and three baseline methods on MOVSW for automating A/B tests, using user clicks logging of news articles from the Addressa Dataset. Our results indicate that MOEA/D is the most stable method with fast convergence compared with other MOEAs, produces more high-quality solutions with high precision than the single-objective methods, and thus is promising to be used in the VCE problem of A/B testing.

© 2023 Elsevier Inc. All rights reserved.

1. Introduction

Online controlled experiments (also known as A/B testing, split tests, or randomized experiments) are widely used to collect implicit user behavior and product change effect of a given change for online and web-facing products, such as social media (Xu et al., 2015), search engines (Tang et al., 2010), social networks (Xu et al., 2015; Feitelson et al., 2013), and web services (Turnbull, 2019; Pajkovic, 2022). The procedure offers different product variants to different groups of users, then collects data related to the user behavior, and compares the different product variants to the unmodified product (Fitzgerald and Stol,

2017; Fagerholm et al., 2017). The A/B test allows the gathering of information for a small but significant percentage of users for stakeholders to make decisions on whether to launch a particular variant to 100% of users (Kohavi et al., 2020; Xu et al., 2015). More specifically, A/B testing is a term for an online controlled experiment with two variants, A and B, which are the control variant (unmodified product) and treatment variant (product modification) of a web or mobile application. The user behavioral, performance, and key metrics are collected, analyzed, and compared during the experiment period to evaluate the effect of the “A” or “B” variant on these metrics. In the ideal case, all the code changes that are pushed to production should be A/B tested first, even though the change may be transparent to the user. This is because some changes may still have an unexpected impact on the user experience and user behaviors.

From the software industry perspective, in recent years, software companies are shifting to embrace the culture of data-driven decision making (Auer et al., 2021; Fabijan et al., 2018b).

[☆] Editor: W. Eric Wong.

^{*} Corresponding author.

E-mail addresses: jjewu@gwu.edu (J.J.W. Wu), mazzu@gwu.edu (T.A. Mazzuchi), sarkani@gwu.edu (S. Sarkani).

With techniques like A/B testing, a large number of hypotheses from product management can be created and evaluated instantly for online services. As a result, A/B testing has been frequently used in product development at web-facing companies, including Facebook, LinkedIn, Pinterest (social media application), Google, Microsoft, Yahoo (information retrieval application), Amazon, eBay, Netflix, Uber, Airbnb (e-commerce application), and others. As an example, the XLNT A/B testing platform in LinkedIn (Xu et al., 2015) instilled A/B experimentation at LinkedIn into the decision-making process and benefited stakeholders, including roles inside or even outside R&D. Similar examples can be found at Microsoft (Kohavi et al., 2013), Facebook (Feitelson et al., 2013), Netflix (Gomez-Urbe and Hunt, 2015) and Google (Tang et al., 2010).

Broadly speaking, the A/B testing process includes two steps:

1. Engineers design and create variants to start the A/B test
2. Stakeholders (or decision makers) make launch decisions based on the A/B test results

However, for both step 1 and 2, it has been noted in Tamburrelli and Margara (2014) that the process for engineers to create variants, start the A/B test and then evaluate the results is manual, time-consuming and error-prone.

The typical lifecycle of A/B experiment is the repeated steps of (1) ideation, (2) design and execution, (3) analysis and learning (Fabijan et al., 2018b). Due to the large number of hypotheses to be validated, and the non-trivial, sometimes unexpected A/B results of various metrics to be analyzed, it is possible that either the optimal hypothesis is not achieved within a single iteration, or these three steps are repeated in several iterations. Therefore, identifying and creating variants tends to be time-consuming, error-prone and tedious activity (Tamburrelli and Margara, 2014). From the impact perspective, more than hundreds of thousands of A/B tests are being conducted each year, with the increasing popularity of A/B testing being adopted by the industry and data-driven software companies (Kohavi and Thomke, 2017; Fabijan et al., 2017). Therefore, addressing these concerns regarding the A/B testing is a valuable undertaking.

With a closer look in step 1, the manual process of engineers designing and creating A/B variants leads to the following two problems:

- It depends solely on the engineers (or designers) in the domain to decide which variants and parameters to be used in the A/B test. Thus, the choice of variants and their parameter values rely heavily on the skills of engineers, who usually have little assistance or guidance in choosing variants.
- It is often not humanly possible to try all combinations of parameter variants in the A/B test to obtain the parameters that lead to the precise optimal A/B test result.

To address the above problems in step 1, a genetic algorithm and aspect-oriented programming is proposed in Tamburrelli and Margara (2014) to support automating the A/B testing (step 1 and 2), by posing it as a search-based software engineering problem. However, the genetic algorithm is based on the assumption that the A/B testing result of each variant has a single metric (objective), represented by a single numeric value such as “o(v)” that indicates the goodness of the variant. Using a single objective overly simplifies the A/B result evaluation, and even if the single objective is composed by a sum of weighted objectives, it is very difficult for engineers to precisely determine the weights for different objectives. In practice, an industrial A/B test result often has multiple metrics (Kohavi and Thomke, 2017). Thus, this single objective genetic algorithm (SOGA) could be very limited when being adopted and used by industry. To the best of our

knowledge, there is no study in literature that automates the A/B testing while considering a multidimensional metric space.

Additionally, the assumption that the goodness of each variant can be described as a numeric value (Tamburrelli and Margara, 2014) not only affects automating the creation of variants in step 1, but also makes the launch decision making in step 2 not applicable to most industrial situations. Very few studies have proposed an analytical approach for the industrial A/B test result with multiple metrics to make launch decisions. A Standard Scorecard Analysis was described based on a company case study (Fabijan et al., 2018a) to let experimenters analyze and examine multiple metrics such as success metrics (also known as Overall Evaluation Criteria) and guardrail metrics, then make decisions on the A/B test in the proposed analysis process. Recently, a Multi-Criteria Decision Making approach to make launch decisions based on multiple metric results was proposed by Wu et al. (2023).

To bridge the literature gap and address the two problems in step 1, in this research, we propose a **Multi-Objective Variant Selection Wrapper (MOVSW)** that utilizes Multi-Objective Evolutionary Algorithm (MOEA) to automate the process of creating and selecting variants as launch candidates in A/B testing. Given a predefined set of parameters to be optimized, the control group in the A/B test uses the existing values of these parameters in the system. The MOVSW automatically creates variants of different parameter values, and conducts A/B testing to evaluate these variants against the control group based on multiple measurements. The outputs of MOVSW are a list of non-dominated candidate variants (also known as Pareto optimal set, or candidate trade-off solutions) that lead to potential improvements on key metrics according to the A/B test results. These candidate variants could be then reviewed by stakeholders to determine the final variant to be launched.

The contributions in this study are two-fold. First, we describe the Variant Creation and Evaluation (VCE) problem in A/B testing, and propose MOVSW that uses MOEA to automate the process of designing, creating, starting, and selecting the variants in A/B testing. MOVSW produces a set of non-dominated variants as launch candidates that aid the final launch decision-making. Second, we designed an illustrative case study on user click logging of news articles, and conducted comprehensive evaluations on MOVSW with several MOEA methods and other baseline SOGA methods in the case study. The case study indicates that MOEA method such as MOEA/D is superior to the baseline SOGA methods.

The remainder of this article is organized as follows. Section 2 introduces the related work. Section 3 introduces the problem statement and formulation. Section 4 introduces the multi-objective evolutionary algorithm. Section 5 is about the design, results, and discussion of the case study. Section 6 provides the conclusion and future work.

2. Related work

The literature review section is split into two parts: (a) A/B testing, (b) multi-objective optimization.

2.1. A/B testing

Historically, the concept of the controlled experiment theory was first introduced in the 1920s by Sir Ronald A. Fisher's experiments at the Rothamsted Agricultural Experimental Station in England (Kohavi and Longbotham, 2017). The controlled experiments can be broadly divided into (1) offline experiments, which have been studied and developed very well in the field of Statistics (George et al., 2005), and (2) online experiments. Online controlled experiment, also known as A/B test, split test,

randomized test, began to gain increasing usage in the late 1990s with the rise of the Internet. With the increasing popularity of web or mobile user-intensive applications, A/B tests have become a gold standard in many internet companies for evaluating new product changes (Xu et al., 2015; Kohavi et al., 2020). Today, many well-known internet companies, such as Amazon, Bing, Facebook, Google, LinkedIn, and Yahoo! etc., run thousands of A/B tests every week to test the effect of changes to the applications. A/B testing is now considered a critical tool (Kohavi et al., 2020) for user-intensive systems, and it has also been used by startups and smaller websites more frequently (Koning et al., 2019).

The academic literature on A/B test started in 2007. Kohavi et al. (2007) initiated the academic discussion in 2007 on A/B test by describing the experience of controlled experiments at Microsoft with guidelines. Later, other well-known tech companies, Facebook (Feitelson et al., 2013) and Netflix (Gomez-Urbe and Hunt, 2015), started to use data-driven decision making (Kohavi et al., 2013), and described their experiences with experimentation to the research community. After more than ten years of research, there has been a couple of works on different problems in this topic, such as (1) the experimentation process definition (Fagerholm et al., 2017), (2) building infrastructure for large-scale controlled experimentation (Xu et al., 2015), (3) metrics selection and development (Machmouchi and Buscher, 2016), and (4) summary of knowledge and challenges in continuous experiments (Auer et al., 2021).

We provide more examples to illustrate challenges in social media, information retrieval, e-commerce and other applications where A/B testing is adopted. Broadly speaking, these applications use A/B testing for the company to make data-driven decisions by listening to the users, rather than the Highest Paid Person's Opinion (HiPPO) (Kohavi et al., 2007). As mentioned earlier, the challenges arise when typically a large number of hypotheses are created and need to be validated via A/B testing on one hand, while the A/B tests being conducted in the industry typically involve multiple metrics in practice on the other hand. An example of the former is the "41 Shades of Blue" A/B tests conducted at Google (Kohavi et al., 2013), where billions of possibilities exist for styling a single product. Another example is the on-demand usage of A/B testing for evaluating the changes or improvements of ranking algorithms and recommendation systems (Hohnhold et al., 2015), where a number of hypotheses or parameters need to be evaluated with potentially multiple iterations. As an example of the latter, the A/B test platform at Microsoft runs analysis to check hundreds to thousands of metrics (Kohavi and Thomke, 2017), and over these years, Bing created more than 6000 metrics, which are grouped by different sub-areas for experimenters to use such as web search, image search, Ads, video search (Kohavi and Thomke, 2017; Fabijan et al., 2017).

2.2. Multi-objective optimization

Multi-objective optimization problems are an integral class of optimization problems in everyday life. These problems normally involve trade-offs, and there is no single optimal solution (Van Veldhuizen and Lamont, 2000). In the case of experiments and decision-making for user-intensive applications, a typical industrial A/B test measures the effect of multiple criteria or multiple key metrics, given these metrics may conflict with each other. For example, the decision makers or stakeholders of an A/B test may wish to maximize benefits from user engagement metrics and operational metrics. However, sometimes generating more content for users would increase the user view rate, but also increase the hide rate in which the user chooses to hide the content, or worsen the latency, a typical key operational metric. With multiple metrics that may conflict with each other, often

the stakeholders would review all possible "trade-off" solutions and choose the one that best satisfies their needs. There are various solutions to address the multi-objective problems (Reddy and Kumar, 2006), such as weighted approach, goal programming approach, interactive approach, etc.

2.2.1. Multi-objective evolutionary algorithms

Over the past 30 years, MOEA has been widely adopted in different areas such as engineering (Fonseca and Fleming, 1998), scheduling (Murata et al., 1996; Tavana et al., 2014; Adibi et al., 2010; Guo et al., 2011; Wang et al., 2018), and data mining (Mukhopadhyay et al., 2013). The first MOEA was produced in mid-1985 (Van Veldhuizen and Lamont, 1998). Since then, a lot of research on the EA-based approach has been done to solve the multi-objective optimization problem. The procedures of MOEA are similar to the genetic algorithm (GA) (Holland et al., 1992; Goldberg and Holland, 1988; Davis, 1991), which was initiated by Holland in 1975. In every iteration of MOEA, elements with better fitness are chosen into the mating pool based on multiple objectives. It subsequently generates new offspring based on operators of reproduction, crossover and mutation in every iteration.

With the development of Search-Based Software Engineering (SBSE) field (Ramirez et al., 2019), metaheuristic search techniques such as GAs (Hamamoto et al., 2018; Oreski and Oreski, 2014; Nazarahari et al., 2019), MOEA (Pradhan et al., 2019; Álvarez et al., 2016; Pascual et al., 2015; Parejo et al., 2016; Ni et al., 2019; Cai et al., 2020; Marcelino et al., 2021; Yeh and Chuang, 2011; Liao et al., 2011; Vignolo et al., 2013; Mirjalili et al., 2016; Macedo et al., 2017; Anagnostopoulos and Mamanis, 2011), have been used in various applications in software engineering (Clarke et al., 2003). However, to our knowledge, there is only one study (Tamburrelli and Margara, 2014) in the literature that uses GA to automate A/B testing, and there is no work that studies the use of MOEA for the problem of automating A/B testing. Tamburrelli and Margara (2014) proposed to use GA to automate A/B testing, with single-objective in the fitness function to represent the goodness of a variant. Although the SOGA is definitely valuable, the A/B testing results in the industry are with multiple objectives in practice. SOGA could be limited since it only considers one objective or a predetermined sum of multiple objectives. When evaluating different variants as it is usually very difficult for decision makers to have a single numeric value to represent the goodness of a variant in the A/B test. The MOEA we propose in this study is capable of providing a set of trade-off solutions to the decision makers. The detailed comparison between SOGA and MOEA is discussed in Section 4.2.

3. Motivating examples and problem statement

In this section, we first describe two motivating examples to illustrate the problem of interest. Then, we summarize the problem statement, referred to as the Variant Creation and Evaluation (VCE) problem.

3.1. Motivating examples

Example 1. We illustrate the main problem with the help of an example, as shown in Fig. 1. Suppose in a web application, the font size of text in production is 20. The stakeholders want to test the effect of changing the font size of the website. To test this requirement, engineers and designers need to decide and create variants of different font sizes to start the A/B test. Finally, let us say font sizes of 14, 16, 24 are selected in 3 variants to conduct the A/B test. Fig. 1 shows typical A/B test results on the 3 metrics for these 3 variants compared with control group. For

A/B Variant Name	Font Size Parameter	Time spent	# user visits	# click through rate
A	14	+2%	+6%	-2%
B	16	+1%	+2%	+1%
C	24	-2%	-3%	-5%
Control	20			

Variants and their parameters in A/B test
 A/B test results

Fig. 1. An example in the web-facing domain to illustrate the Variant Creation and Evaluation problem in the A/B testing process.

A/B Variant Name	Special Effects Filter Parameter	Color Correcting Filter Parameter	Time spent	# photos taken	# setting changes
A	1.0	0.8	+10%	-2%	+5%
B	1.0	0.2	+9%	-3%	+3.6%
C	0.2	0.8	+3%	+7%	+3.4%
D	0.2	0.2	+3.6%	+4%	+0.2%
Control	0	0.5			


Variants and their parameters in A/B test for digital cameras (embedded systems) 
A/B test results

Fig. 2. An example in embedded system domain to illustrate the Variant Creation and Evaluation Problem in the A/B testing process.

each metric, % change of the metric on the variant over control group is displayed in the table. Based on the results, variant C is a clear no-go, since all of the three metrics in group C appear to be worse than the control group. On the other hand, variants A and B could be potential launch candidates, but it may be hard to determine which one is the winner (A and B are trade-off candidate solutions). Specifically, A has a huge +6% improvement on “#user visits”, much higher than +2% of B, however, A also shows a -2% drop in “# click through rate” that could be hurtful. In this case, A and B would be both selected as the candidate variants, and they are sent to the stakeholders and decision makers in a discussion to choose one variant to be launched. One potential outcome of the discussion could be: variant A is chosen as the final launch candidate because “# click through rate” is not a top tier metric and the stakeholders think it is fine to have a -2% regression on this metric in order to realize +6% gain in top metric “# user visits”.

Example 2. Besides the above example in the web-facing domain, from Fig. 2, we provide another example in the embedded system domain, where A/B testing is being adopted, with challenges and problems reported (Olsson, 2018). Suppose the requirement is to conduct A/B test in digital cameras, one example of embedded systems. The goal is to evaluate the effect of combining a new special effects filter with a modified color correcting filter. In the existing digital cameras (control group), there is no special effects filter and the default parameter in the color-correcting filter is 0.5. Suppose the variants with different values for the two parameters (one for special effects filter, one for color-correcting filter) are selected for conducting the A/B test, and the corresponding A/B test results are displayed in Fig. 2. From the results, B would be dropped since it could be dominated by A. A, C and D would

form the non-dominated set (or Pareto optimal set) and be sent to stakeholders for making a decision. One possible decision could be that D is selected finally to be launched: “Time spent” and “# photos taken” are two business-critical metrics, so the more positive, the better. However, “# setting changes” is a guardrail metric (Deng and Shi, 2016), and having this metric > +2% is a hard no for it to be launched, so A and C are both rejected.

3.2. Problem statement: The variant creation and evaluation (VCE) problem

As described in the examples, assuming that the A/B metrics are selected and defined by stakeholders already, a key question towards automating A/B test is: Rather than relying on domain experts, how do we automate the process of designing, creating, evaluating and selecting the variants via A/B testing process, to produce a set of high-quality candidate variants to aid the final launch decision-making? To answer the question, we define the Variant Creation and Evaluation (VCE) problem that comes across in the A/B testing process, from the early design phase to the final evaluation phase. VCE problem is about (1) deciding which kinds of variants to create and test in the A/B testing, and (2) evaluating and selecting a set of high-quality candidate variants to aid the final launch decision makings. The VCE problem can be addressed by different solutions, in which the output of the solution is a set of candidate variants. Given the final launched variants as ground truth, various measurements such as recall and precision can be calculated for the output of each VCE solution to evaluate the quality of the solution.

The VCE problem has a significant influence on the design and final decision making of the A/B test. However, as illustrated in the examples, the current manual approaches to the creation and

selection of variants are subjective, and depend highly on the team members' domain knowledge, decision making, experiences and judgment. Nevertheless, we argue that search-based technique such as MOEA is a well-qualified method to address the VCE problem. Specifically, search-based techniques provide solutions for developers in finding good variants and thus moving toward more automated A/B tests, supported by the 4 reasons (Clarke et al., 2003; Bowman et al., 2010) below:

1. It has a very large search space
2. There is no known efficient and complete solution
3. One can define suitable fitness functions
4. Generating candidate solutions should be inexpensive

4. MOVSW: The proposed VCE solution based on MOEA

In this section, we introduce the proposed MOVSW that utilizes MOEA to address the VCE problem in A/B testing. Fig. 3 illustrates visually our proposed 2-step approach towards automating the A/B testing process: it includes addressing the VCE problem using the proposed MOVSW in step 1, and launch decision making in step 2.

To provide more background and context, we note that the proposed MOVSW is carefully designed so that the basis and motivations do not lose generality and are based on both of (1) the already established A/B testing fundamentals (Kohavi et al., 2020; Wu et al., 2023) and (2) published empirical examples of A/B testing usage in industry (Tang et al., 2010; Xu et al., 2015; Kohavi and Thomke, 2017). Based on the above references, we observed the literature gap that there is no method that automate the creation and selection of A/B testing, where the A/B results include multiple metrics in practice. Based on this observation, we created the VCE problem, and designed MOVSW to address it.

4.1. Formulation of A/B testing, VCE problem and VCE solution

Mathematically, suppose \vec{c} is a D -dimensional vector that represents the existing values of D parameters of interest in the system. \vec{c} is used in the control group (unmodified product) of the A/B testing, and in this work, we assume any of these D values can be optimized within certain constraints such as upper and lower bounds. The goal of A/B testing is to find new values for these parameters that lead to improvements in the goal metrics when comparing treatment group with control group. In other words, A/B testing attempts to find a variant \vec{x} for the multi-objective optimization problem:

$$\max\{f_1(\vec{x}, \vec{c}), f_2(\vec{x}, \vec{c}), \dots, f_n(\vec{x}, \vec{c})\}$$

where \vec{x} is a D -dimensional variable vector representing D parameters of the variant in the system; $f_i(\vec{x}, \vec{c})$ is the result of metric i in the A/B test result of a given variant \vec{x} . It is formally defined in Section 4.3, representing the relative change for the metric i between the control group (with old parameters \vec{c}) and the treatment group (with new parameters \vec{x}) in the A/B test result. However, in A/B testing, it is common to see A/B metrics (objectives) conflict with each other. So, if we want to find a single globally optimum solution to the multi-objective optimization problem, it would be an NP-complete problem. Nevertheless, the VCE problem in the A/B test process aims to find a set of trade-off candidate variants $X' = \{\vec{x}\}$ for this multi-objective optimization problem. Instead of finding a single variant as the solution, having a set of candidate variants as output has the benefit that distinct variants can speak for various trade-offs from these objectives, and are near to the optimal solution.

Given the formulation of VCE problem, MOEA provides a well-qualified solution to this problem, because MOEA outputs a population of trade-off solutions instead of a single solution. The

proposed method MOVSW takes MOEA into consideration. By using MOEA, we can obtain a series of non-dominated launch candidates for VCE problem and then let the decision makers choose the final candidate(s) according to actual requirements and feedback. For example, we may prefer one candidate with the best A/B result in metric X, or prefer a candidate that does reasonably well in metric Y without hurting metric Z too much.

4.2. Single objective approach using sum of weighted objectives

Besides solving the problem using MOEA, another alternative is the single objective approach to sum the values of multiple objectives with weights. However, the weights are usually very subjective and problem dependent (Bowman et al., 2010), and it is hard for decision makers to set, adjust and normalize these weights. Besides, the definition, scale and priority of metrics in A/B test may also change over time (Deng and Shi, 2016). Furthermore, for our problem in A/B testing, the launch decision-making process of A/B test result (i.e., selecting variant to launch) is empirical and involves discussions and evaluations among domain experts (Tang et al., 2010; Xu et al., 2015), rather than using a weighted sum of objectives. The MOEA approach evaluates fitness based on multiple objectives and outputs a set of Pareto front solutions, so it does not need normalization or weights since objectives are compared individually. Therefore, we believe that the range-independent, vector-based MOEA approach better fits the A/B test use case, and let the decision makers decide the final solution from trade-off candidate solutions, as described in Section 4.7.

For a more clear comparison, we summarize two advantages of multi-objective algorithms over single-objective algorithms:

1. Multi-objective algorithm decouples the scoring of variants, i.e., selection of the "best" variants, out of the generation of the candidate variant set. The decoupling characteristic indicates that:
 - (a) Pareto set of the multi-objective algorithm allows more flexibility in selecting the suitable variant(s) from a candidate set pool. Specifically, rather than using fixed assigned weighted sum, other methods can be used such as the manual method with the domain experts' expertise & knowledge.
 - (b) Changes or updates in the scoring of variants, which might happen frequently, do not affect the generation of candidate Pareto set for multi-objective algorithm.
2. Pareto set generated by multi-objective algorithm has higher quality, compared with single-objective algorithm. There are mainly two reasons are listed below:
 - (a) The candidate variant set from single-objective algorithm may include more "useless" variants that are dominated by other variants. These "useless" variants potentially reduce the quality of the candidate set. This will be validated in our experiments in Section 5.4.2.
 - (b) As mentioned in 1., the candidate variant set from single-objective algorithm heavily depends on the method to select the "best" variants. If the method to select variants is inaccurate or error-prone, the quality of the candidate set would suffer. Multi-objective algorithm does not have this issue.

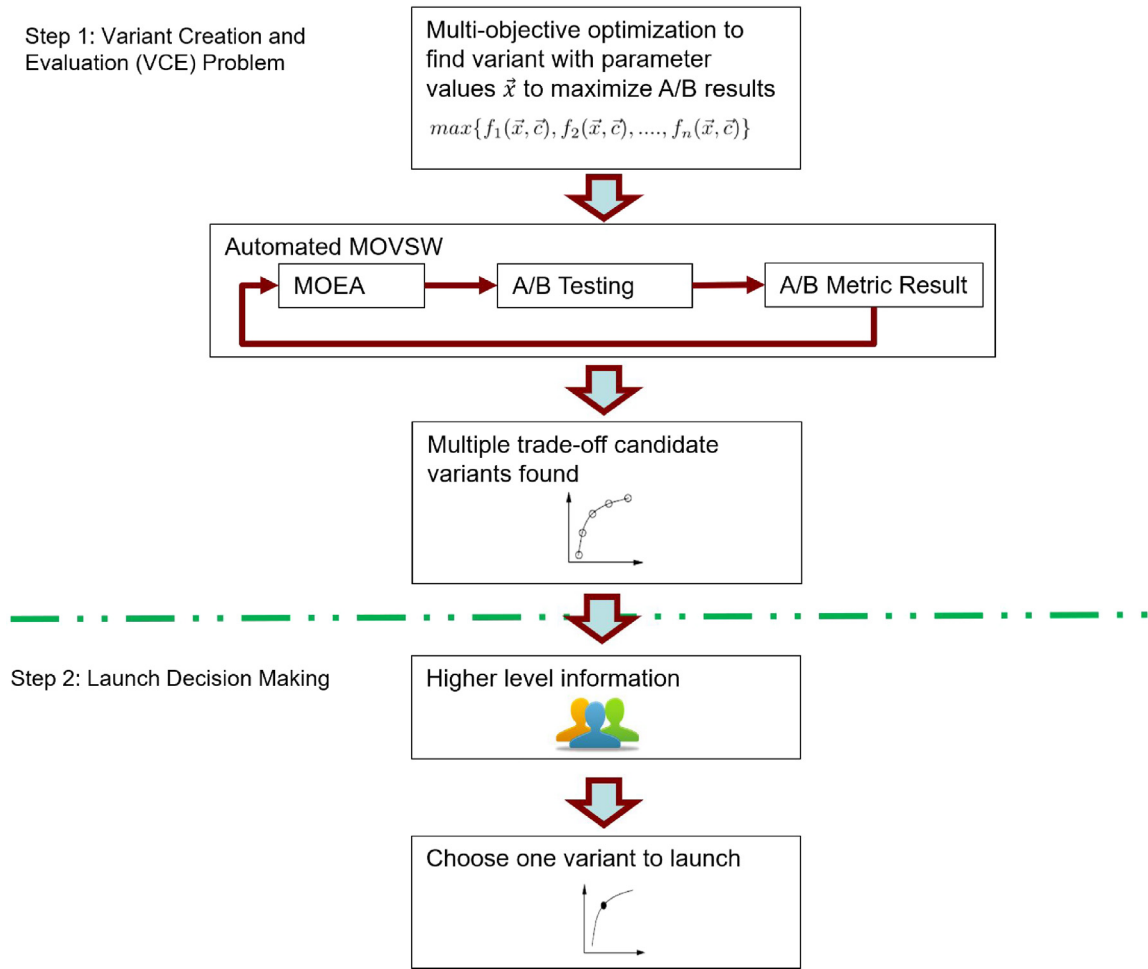


Fig. 3. 2-step approach towards automating the A/B testing process: it includes addressing the VCE problem using the proposed MOVSW in step 1, and launch decision making in step 2.

4.3. Chromosome representation & fitness function

We describe the encoding schema, i.e., chromosome representation, and the fitness function of the chromosome. Suppose we need to perform A/B test on D parameter variables in the software system, we convert and encode these D variables into a feasible solution $\tilde{x} = \{x_1, x_2, \dots, x_D\}$, an array of D real numbers, where \tilde{x} is restricted by the lower bound $\tilde{L} = \{l_1, l_2, \dots, l_D\}$ and the upper bound $\tilde{U} = \{u_1, u_2, \dots, u_D\}$ of variables. Given n metrics in A/B test, the fitness functions $f = \{f_1(\tilde{x}, \tilde{c}), f_2(\tilde{x}, \tilde{c}), \dots, f_n(\tilde{x}, \tilde{c})\}$ are defined as the percentage change (or relative change) for each metric between the control group (with old parameters \tilde{c}) and the treatment group (with new parameters \tilde{x}) in the A/B test result. Specifically, for the i th metric in A/B test, the fitness function is

$$f_i(\tilde{x}, \tilde{c}) = \text{normalize}((m_{i,t}(\tilde{x}) - m_{i,c}(\tilde{c}))/m_{i,c}(\tilde{c}))$$

where $m_{i,t}$ and $m_{i,c}$ is the raw value of the i th metric for treatment \tilde{x} and control \tilde{c} respectively. Note that to simplify processing of raw metrics from A/B test result, we assume the percentage change of metric is normalized, flipped and scaled, represented by *normalize* function, so that

1. $f_i(\tilde{x}, \tilde{c})$ is set to 0 if it is not statistically significant after the multiple testing adjustment (commonly used to adjust metric result in A/B test).
2. Positive $f_i(\tilde{x}, \tilde{c})$ means it has positive benefits for goals (or success of the experiment).

4.4. Population initialization, crossover and mutation

We now describe the algorithm of MOVSW by utilizing MOEAs in Algorithm 1. We first use *initialization()* to randomly generate P chromosomes within the upper and lower bounds as an initial population. In the next step (step 4 in Algorithm 1), we apply the crossover operator, and mutation operator in MOEA and then produce new chromosomes with population size P for each iteration, represented by *getNewPop()* function. For the crossover operator, two chromosomes are selected at random following the crossover probability, and are transformed into two new chromosomes with the crossover operation. For the mutation operator, a chromosome is selected following the mutation probability, and is transformed to a new chromosome with the mutation operation. In our method, we use the Simulated Binary Crossover (Deb et al., 1995) as the crossover operator. As for the parameters, we set the probability of crossover as 100%, and set the distribution index of simulated binary crossover as 20. We use the Polynomial Mutation (Deb et al., 1995) as the mutation operator. As for the parameters, we set the expectation of the number of mutated variables as 1, and set the distribution index of polynomial mutation as 20.

4.5. Recombination

Then, we combine the new chromosomes with the current Pareto optimal set and compute the new non-dominated chromosomes in PO_{i+1} as the new Pareto optimal set. In other words,

Table 1
Summary of MOEA and SOGA algorithms evaluated in this research.

Algorithm	Category	Description	Reference
NSGAII	MOEA	Non-dominated sorting genetic algorithm II	Deb et al. (2002)
SPEA2	MOEA	Strength Pareto evolutionary algorithm 2	Zitzler et al. (2001)
PESAII	MOEA	Pareto envelope-based selection algorithm II	Corne et al. (2001)
MOEA/D	MOEA	Multiobjective evolutionary algorithm based on decomposition	Zhang and Li (2007)
AGEII	MOEA	Approximation-guided evolutionary multi-objective algorithm II	Wagner and Neumann (2013)
GA	SOGA	Genetic algorithm	Holland et al. (1992)
DE	SOGA	Differential evolution	Storn and Price (1997)
SA	SOGA	Simulated annealing	Bertsimas and Tsitsiklis (1993)

to ensure elitism, all the old and newly generated chromosomes are combined and then passed for non-dominated sort again. The reason for this is that from the experiments, we found this accumulated Pareto optimal set has more accurate and more stable results than using the Pareto optimal set from each iteration.

Algorithm 1: Pseudocode of MOVSW by Utilizing MOEA

Input: Size of parameters D , Parameter values in control group $\vec{c} = \{c_1, c_2, \dots, c_D\}$, Fitness functions $\vec{f} = \{f_1, f_2, \dots, f_n\}$ for n metrics in A/B test, Population size P , Iteration number T
Output: Pareto Optimal Set as candidate variants $\{x^*\}$

```

1:  $i \leftarrow 0$ 
2:  $PO_i \leftarrow \text{initialization}(P)$ 
3: while  $i < T$  do
4:    $PN_i \leftarrow \text{getNewPop}(PO_i, \vec{f})$ 
5:    $U_i \leftarrow PO_i \cup PN_i$ 
6:    $PO_{i+1} \leftarrow \text{getParetoOptimalSet}(U_i)$ 
7:    $i \leftarrow i + 1$ 
8: end while
9: return Pareto Optimal Set  $PO_i$ 

```

Algorithm 1: Algorithm of MOVSW by Utilizing MOEA

4.6. MOEA optimizer selection

To keep the wrapper as generic as possible, any MOEA optimizer can be plugged into the MOVSW in theory. In implementation, we choose MOEA/D algorithm as the default MOEA optimizer for MOVSW given the superior and more stable results of MOEA/D in our case study evaluations. In addition to MOEA/D, we also used and evaluated several other MOEAs for MOVSW. Table 1 lists the MOEAs that are used and evaluated in the case study. For comparison in the experiment, it also includes the SOGAs. In the case study, we evaluated these algorithms based on the implementation from PlatEMO (Tian et al., 2017).

4.7. Choosing single solution from MOSVW output

Given the Pareto optimal set, a list of candidate variants as output of the MOSVW, the final decision is left with decision makers or stakeholders to discuss and make launch decisions. This launch decision-making process of A/B test result is empirical and involves discussions and evaluations among domain experts (Tang et al., 2010; Xu et al., 2015). In web-facing companies, it also involves alerting, scorecards, and periodical diagnosis on different types of metrics in A/B test results in practice (Wu et al., 2023). Since this process is currently empirical, our focus is on the overall quality of Pareto optimal set from MOSVW (the output of

VCE Problem in Step 1 of Fig. 3), rather than the evaluation of the empirical process of selecting the final launch candidate from the MOSVW output (the launch decision making in Step 2 of Fig. 3).

5. Case study

In this section, in order to validate our approach for a VCE problem, we performed a comparative case study to apply various MOEAs and other baseline methods on the MOVSW to selecting A/B variants that optimize the goal metrics of a news recommendation algorithm. Given a news recommendation algorithm with the existing parameters and their values, we focus on evaluating the use of MOVSW on selecting variants with new parameter values that yields better A/B test result. The A/B test results in this experiment are obtained from the user logging data in the Adressa dataset (Gulla et al., 2017). Section 5.1 describes the main research questions in this case study. Section 5.2 introduces the Adressa dataset used in the experiment. Then, the experiment setting is discussed in Section 5.3. Finally, Sections 5.4 and 5.5 involve the results and the corresponding discussions of the case study.

5.1. Research questions

The experiments we conducted aim at answering the following research questions:

RQ1: (Effectiveness) To what extent can the proposed MOEAs produce candidate variants that better optimize the A/B test metrics in a VCE problem?

RQ2: (Comparison to single-objective algorithms) How do MOEAs perform compared with SOGAs in a VCE problem?

RQ3: (Effects of MOEA parameters) What are the main factors affecting the evaluation results of MOEAs in a VCE problem?

The goal of RQ1 is to validate that MOEA is effective in recognizing good trade-off solutions for the VCE problem. RQ2 mainly targets demonstrating that MOEA is superior to other SOGA alternatives. The goal of RQ3 is to test the effects of different population sizes, because this is one of the most critical parameters in MOEA.

5.2. Dataset

Gulla et al. (2017) published the Adressa dataset, formed by the logs of the Adresseavisen website in ten weeks. Overall, it includes 48,486 news articles, 3,083,438 users and 27,223,576 click events. The click events include attributes such as session time, news title, news category and user ID. The news articles in the dataset are in Norwegian, and are connected with information such as authors, entities and body. We use the Adressa dataset in this case study for constructing the A/B metric results of a given variant.

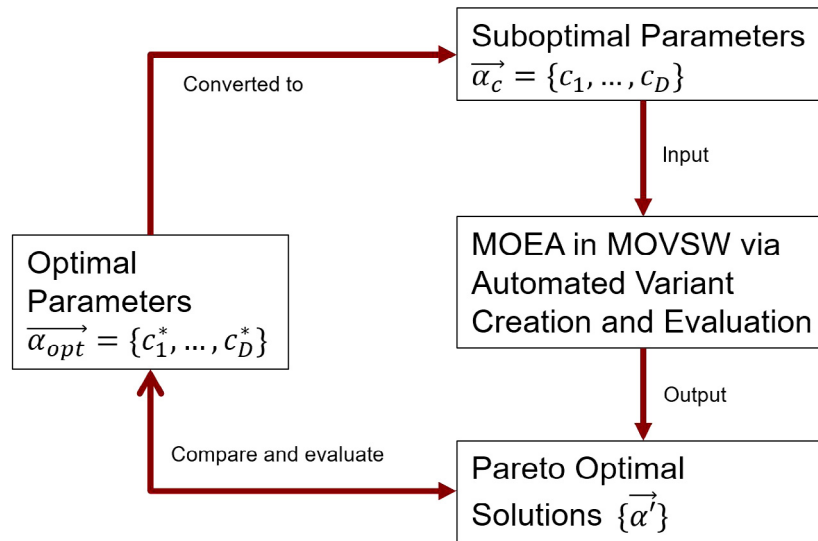


Fig. 4. The Evaluation methodology for the case study using Adressa Dataset. The size of parameters D is 3 in the case study.

5.3. Experiments setting

We first describe the evaluation methodology of the experiment. Fig. 4 provides the visual illustration of the evaluation methodology. To evaluate the effectiveness of MOVSW, we let the control group in MOVSW take the sub-optimal parameters of a news recommendation algorithm. Then we apply MOEA to improve upon the sub-optimal parameters towards optimal parameters (ground truth) via automated variant creation and evaluation in MOVSW. The goal is to evaluate the quality of the output variants of the MOVSW by either (1) checking how close the parameters in the variant are to the optimal parameters as ground truth, or (2) checking how the output variants improve the objectives. This validates whether using MOEA to modify certain parameters of the news recommendation algorithm as variants in the A/B test can indeed result in positive A/B results that improve the goal metrics once launched.

A maximum of 100 generations are chosen as the stopping criteria of the MOEA optimization in MOVSW. A population size of 100 is selected for all experiments except RQ3. MOVSW uses the Simulated Binary Crossover (Deb et al., 1995) as the crossover operator. We set the probability of crossover as 100%, and set the distribution index as 20. Polynomial Mutation (Deb et al., 1995) is used as the mutation operator. We set the expectation of the number of mutated variables as 1, and set the distribution index of polynomial mutation as 20. In the experiments, the total number of the decision variables D is 3 (also known as the size of parameters D in MOVSW input in Algorithm 1), and they are assigned random values during the MOEA initialization as sub-optimal parameters. Three objectives are used as the fitness functions of MOVSW:

1. Active time spent on the news article in Android (OS).
2. Active time spent on the news article in Windows (OS).
3. Active time spent on the news article in Mac (OS).

The fitness function of the MOEA algorithm for metric (objective) i is defined as the % change of m_t over m_c , where $m_t = activeTime_{gt} - |activeTime_{predicted} - activeTime_{gt}|$ is the metric i result for treatment group, and m_c is the similar metric for control group; $activeTime_{predicted}$ and $activeTime_{gt}$ represent the active time metric from the recommendation model prediction and the actual active time in the Adressa dataset respectively. Note that

m_t is lower if the predicted value has a larger difference than the actual value.

A first set of experiments were conducted with five different MOEAs for RQ1. Another set of experiments was conducted with single-objective optimization, i.e., three different SOGAs for RQ2. Since SOGA only has one objective and fitness function, we use the same setting as MOEA experiments except that the fitness function is defined as the sum of weighted objectives for the three metrics used in MOEAs, with different sets of weights being evaluated. A third set of experiments tested the use of different population sizes in MOEA for RQ3. Table 1 lists the MOEAs and SOGAs (as baseline methods for comparison) that are used and evaluated. The result analysis is presented in Section 5.4. The implementation details of the evaluation are described and discussed next.

News Recommendation Algorithm and Optimal Parameters. To set up the evaluation, we first chose a news recommendation algorithm, and calculated the optimal parameters for the news recommendation algorithm from the user logging data in Adressa dataset that maximizes the metrics. Our criteria for choosing the news recommendation algorithm for the experiments are as follows: (1) the algorithm needs to be simple to make the experiment easy to reproduce; (2) it does not have too many parameters and changing the parameter values should cause changes in the algorithm output. Based on these criteria, we implemented a simplified version of the Decaying Model in Okura et al. (2017) as our news recommendation algorithm in the experiment. Given a news article represented as a vector a , the model calculate the relevance score $R(u_t, a) = u_t^T a$, where $u_t = F(a_1, \dots, a_t)$ is the user state, depending on previously viewed articles a_1, \dots, a_t . This score indicates the measurement of the targeted metric (objective) such as the user's rating, or time spent on the article, predicted by the model. In the simplified model, for the objective i among the total n objectives, the user state $u_{ti} = F(a_{1i}, \dots, a_{ti}) = \alpha_{1i}a_1 + \alpha_{2i}a_2 + \dots + \alpha_{ti}a_{ti}$, where $\alpha^{(i)} = [\alpha_{1i}, \dots, \alpha_{ti}]$ is the model parameters. So the score for objective i is $R(u_{ti}, a) = u_{ti}^T a = \alpha_{1i}a_1 + \alpha_{2i}a_2 + \dots + \alpha_{ti}a_{ti}$. Finally, $\alpha = [\alpha^{(1)}, \dots, \alpha^{(n)}]$, are the total parameters of the multi-linear news recommendation model.

The optimal parameters for the recommendation algorithm are also needed in order to know how far the parameters from a given variant are close to optimal. To achieve this, we train the model from the training data of the user click events to obtain

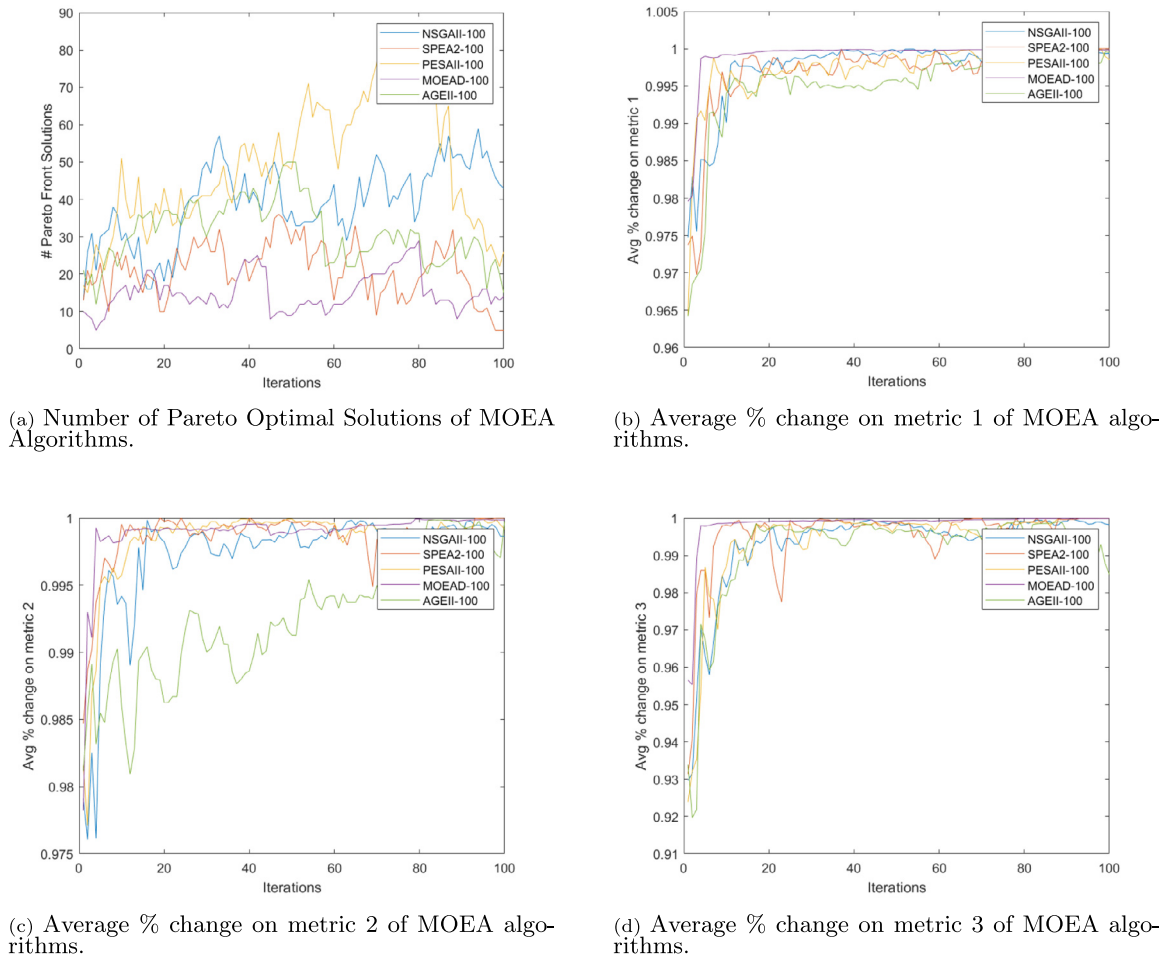


Fig. 5. Number of Pareto optimal solutions and average % change on metric 1 of MOEA algorithms.

the optimal parameters α_{opt} that minimize the loss of the multilinear model for the training data. We selected 3 parameters from them as the decision variables of the MOEA and changed them to random values as MOEA input, and they are to be optimized in the MOEA optimization.

Preprocessing for User Click Data and News Articles. We describe the preprocessing of the data and the use of them in the experiments. We downloaded the full Adressa News Dataset which include 10 weeks of user client events and corresponding news articles that users viewed. We did several filtering on the user click events: We first filtered the user click events in which the news article information is unavailable in the dataset. We then filtered the click events that have higher than 0 “active-Time”, and also did filtering to include the only users that have over 500 click events. Finally, the data were split into training set and testing set, which were used by computing optimal parameters and the MOEA experiment respectively. For the users in the testing data, we randomly split them into two groups: group A and group B, for the A/B test evaluation in the MOEA experiments.

5.4. Results

5.4.1. RQ1: Assessing the effectiveness of MOEAs

Fig. 5(a) shows the number of Pareto optimal solutions of five MOEAs in each iteration. As aforementioned, the population size is 100 for all algorithms. The fitness functions are defined as the % change of treatment over control on these three metrics

respectively: active time spent on OS Android, Windows and Mac. The number of Pareto optimal solutions is not very stable for all of the five MOEAs. Among them, MOEA/D seems to be the most stable and the number of Pareto optimal solutions is mostly 10–25. PESAI is the most unstable algorithm as the number is around 80 at iteration 80, and dropped to around 25 at iteration 100. In the first iterations, e.g. the 10th iteration, the number of Pareto optimal solutions already reaches 12–50. With the analysis below, we can infer that the number of Pareto optimal solutions does not necessarily correlate with the quality of the solutions.

Figs. 5(b), 5(c) and 5(d) shows the average percentage change of the five MOEAs per iteration on metric 1, metric 2, and metric 3 respectively. Note that the upper bound of the % change of metrics is 1 due to its definition. We can see that except for AGEII, other MOEAs (NSGAII, SPEA2, PESAI, MOEA/D) perform pretty well, with % change close to 100%. Among them, MOEA/D rises to a stable state at only iteration 6–8 for all of the three metrics, and thus converges faster than all the other MOEAs. Also, MOEA/D is the most stable curve compared with other curves. This could be due to that the decomposition method of MOEA/D has the advantage of producing a set of very evenly distributed solutions (Zhang and Li, 2007). AGEII performs the worst for metric 2, and performs slightly worse than other MOEAs for metric 1. Although slightly different curves are observed for different runs due to the randomness of MOEAs, the steady increase of the percentage change for all five MOEAs demonstrates the effectiveness of the MOEAs. In other words, the MOEAs are

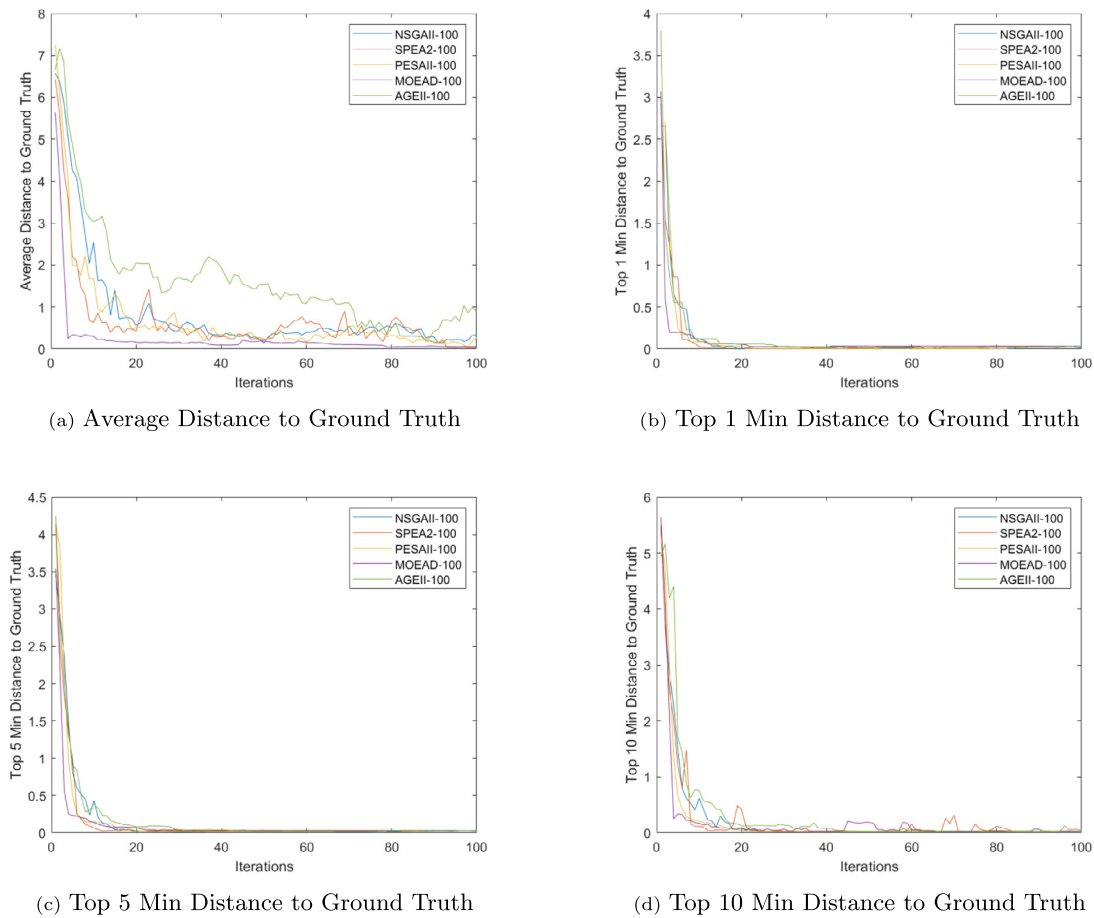


Fig. 6. Distance of Pareto Optimal Solutions to Ground Truth of different MOEA algorithms.

truly optimizing the fitness functions towards maximizing the percentage change of MOEA on the three metrics individually.

Besides the percentage change of metrics, the distance between the Pareto optimal solutions and the ground truth is another way to measure the performance of MOEAs. Fig. 6(a) shows the average distance between the Pareto optimal solutions and the ground truth, defined as the optimal parameters of the recommendation model. Similar to the curves in the above percentage change of metrics, we can observe that other than AGEII, all the rest MOEAs (NSGAII, SPEA2, PESAI, MOEA/D) have a trend of steep decrease to close to 0 with more iterations. AGEII is still the worst, with an average distance of 1 at iteration 100, while all the other MOEAs reach < 0.5 .

We further compute the minimal distance for the top k ($k = 1, 5, 10$) Pareto optimal solutions, rather than the average distance for all Pareto optimal solutions. This can show the distributions of good solutions in the Pareto optimal solutions, while the average distance might be skewed by the particularly bad ones from Pareto optimal solutions. Figs. 6(b), 6(c) and 6(d) show the distance between the ground truth and the top 1, 5 and 10 Pareto optimal solutions with minimal distances. From the result, all of the five MOEAs converged and reached less than 0.2 distance after iteration 30. Among them, MOEA/D is still the most stable one and achieves the lowest distance in early iterations, compared with other algorithms. From the convergence shape of the curves, the top 5 Pareto optimal solutions are reasonably good solutions given the stable curve and fast convergence speed. The solutions between the top 5 and top 10 are lower quality solutions since some up and downs in the curves are observed in Fig. 6(d).

Table 2

Evaluation result on average and top 1 distance to ground truth for MOEAs at the last iteration (iteration 100).

Algorithm	Category	Average distance	Top 1 Min distance
NSGAII	MOEA	0.1046	0.0408
SPEA2	MOEA	0.5520	0.0331
PESAI	MOEA	1.1572	0.0448
MOEA/D	MOEA	0.0238	0.0438
AGEII	MOEA	0.5919	0.0994

To give the numerical comparison, Table 2 shows the average distance and top 1 minimal distance between the ground truth and various types of MOEAs and SOGAs at the last iteration (iteration 100). Among all the evaluated algorithms, MOEA/D, NSGAII and SPEA2 achieve the superior average distance to ground truth compared with other algorithms. In terms of top 1 minimum distance, MOEA/D, NSGAII, PESAI and SPEA2 achieve similar results, while AGEII gets much worse results. This indicates that the top 1 best solutions from Pareto optimal solutions for each of the four MOEAs are on a similar level.

Summary for RQ1: In terms of the Pareto optimal solution set, MOEA/D is the most stable algorithm with the fastest convergence among the five MOEAs. AGEII is unstable with the worst performance. For the top solutions from Pareto optimal solution set, MOEA/D, NSGAII, PESAI and SPEA2 are comparable, validating the hypothesis that they can effectively produce variants that improve the A/B test metrics, and AGEII is the worst.

Table 3

Evaluation result on the average number of with-in range solutions for MOEAs and SOGAs at the last iterations.

Algorithm	Category	# With-in Range Solutions (Average of last K iterations)				
		$K = 5$	$K = 20$	$K = 30$	$K = 50$	$K = 100$
MOEA/D	MOEA	30.0	28.3	26.6	22.1	15.6
AGEII	MOEA	13.0	13.6	13.0	11.9	7.2
NSGAI	MOEA	13.0	21.8	22.5	21.7	15.6
PESAI	MOEA	18.6	20.6	20.3	19.3	12.9
SPEA2	MOEA	28.2	24.8	24.3	22.6	16.7
GA	SOGA (Weighted Sum)	22.6	20.7	18.4	18.6	14.6
DE	SOGA (Weighted Sum)	15.0	12.4	10.6	10.3	5.7
SA	SOGA (Weighted Sum)	0	0	0	0	0

5.4.2. RQ2: Comparison with single objective genetic algorithms (SOGAs)

The goal of RQ2 is to determine if there are benefits from using MOEA when compared to simpler search heuristics such as SOGA. We define the single-objective fitness function as the sum of weighted objectives for the three metrics, which are used in MOEA for the case study. The weights are defined as follows:

- w_1 : weight for metric “Active time spent on the news article in Android (OS)”.
- w_2 : weight for metric “Active time spent on the news article in Windows (OS)”.
- w_3 : weight for metric “Active time spent on the news article in Mac (OS)”.

Given the three objectives, two sets of weights are evaluated in SOGA. The first is $w_1 = w_2 = w_3 = 1/3$, representing equal weights. The second is $w_1 = 1, w_2 = w_3 = 0$, meaning that the single fitness function only represents % change of treatment over control on the metric: “Active time spent on Android (OS)”.

Evaluation Metrics. To evaluate the overall performance of the output to VCE problem from each algorithm, for either MOEA or SOGA, we evaluate the number of “good” candidate variants from the Pareto set of candidate variants produced by the algorithm, named as “# with-in range solutions”. The “good” candidate variant is defined as the variant whose distance to the ground truth is less than $L(L = 0.05)$. To summarize, two metrics are used to evaluate the quality of each algorithm:

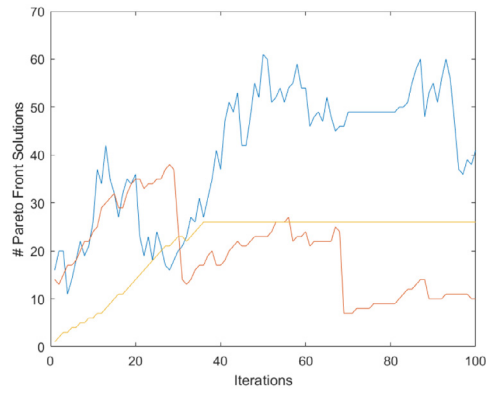
- **#With-in range solutions.** The number of with-in range solutions captures how many relevant variants are produced by an algorithm of interest. Note that the population size is set as 100 for either MOEA or SOGA, given that the population size is one of the main input parameters of the algorithms, as shown in Table 5. This ensures that the number of raw outputs from SOGA or MOEA per iteration is at most 100. The Pareto set is then extracted from the raw outputs of the algorithms. The Pareto set is used instead of raw outputs because only solutions from Pareto set represent meaningful solutions in the VCE problem of A/B testing. Therefore, it is fair to compare the absolute number of with-in range solutions given that the raw output at each iteration of either SOGA or MOEA is capped at 100 by the algorithm parameter of population size.
- **Precision: The percentage of with-in range solutions in Pareto set.** This percentage captures the quality of Pareto set produces by each algorithm. Higher percentage represents that the algorithm returns more relevant (with-in range) solutions and returns less irrelevant solutions in the Pareto set.

Note that another useful metric besides the above two metrics is *Recall*. However, the recall metric could not be computed since the number of all relevant with-in range solutions is unavailable in this experiment.

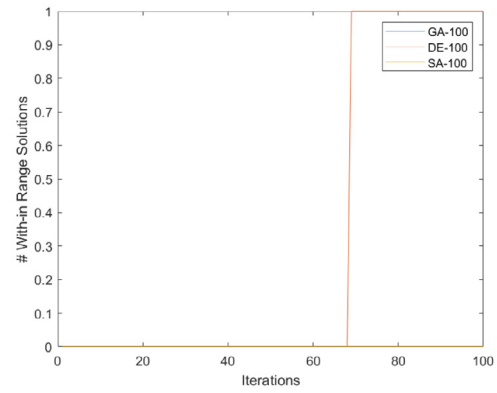
Result Discussion on #With-in Range Solutions. Before diving into the results, we have validated in all experiments of this case study that, the raw output size per iteration for either SOGAs or MOEAs is not greater than 100 since the parameter of population size is set as 100 for each algorithm. We validated this to ensure the fairness of comparing SOGAs with MOEAs in terms of the absolute number of with-in range solutions.

Fig. 7 shows the number of Pareto optimal solutions and with-in range solutions in each iteration for three sets of algorithms: (1) SOGAs on one metric, with weights $w_1 = 1, w_2 = w_3 = 0$, (2) SOGAs on the weighted sum of three metrics, with weights $w_1 = w_2 = w_3 = 1/3$, and (3) MOEAs on three metrics individually. Note that the same parameter setting is used for all algorithms (e.g. the population size is 100 for all algorithms). We observe a higher number of Pareto set from MOEAs when comparing the Pareto set volume of SOGAs from Fig. 7(c) and MOEAs from Fig. 7(e). This is reasonable because MOEA naturally produces more non-dominated solutions than SOGA. Interestingly, from Fig. 7(a) and 7(c), GA produces more Pareto solutions in (1) than in (2), but the quality is much worse in (1), as shown in the with-in range results. This aligns with our finding in RQ1 that the size of Pareto set does not have a correlation with the quality of the algorithms. From Fig. 7(b), for SOGAs on one metric in (1), there is 0 with-in range solution identified in SA and GA, and only 1 with-in range solution in DE. This means that SOGA with fitness function only optimizing one metric works poorly in producing high-quality output. This is expected because the other two different objectives are missing in the SOGA computation. From Fig. 7(d), for SOGAs on weighted sum of three metrics in (2), we observe much better with-in range solutions for DE and GA than (1). This shows that assigning equal weights turns out to be a good approximation for this particular case study at least for GA and DE. GA, the classic algorithm, has the most with-in range solutions even in early iterations, compared with DE and SA. SA still has 0 with-in range solutions with equal weights. With-in range solutions start to pop out for DE in iteration 40, and go up steadily with more iterations. From Fig. 7(f), MOEA/D and SPEA2 have the most with-in range solutions, better than all the three SOGAs (GA, DE, SA) and the other MOEAs (PESAI, AGEII, NSGAI). The main reason is that the Pareto set produced by SOGA has a much smaller size than MOEA, which is a major key advantage of MOEA over SOGA. This limits the size of with-in range solutions that SOGA can produce, compared with the best-performing MOEAs (MOEA/D and SPEA2 in this experiment).

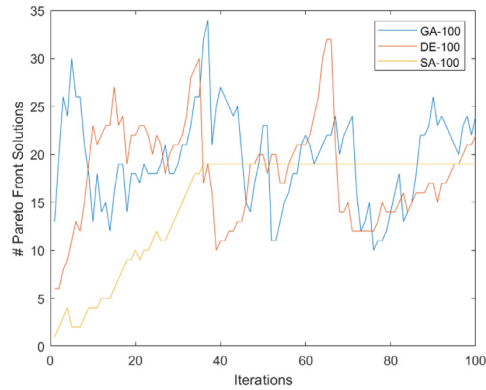
Furthermore, we checked the numeric results of MOEAs and SOGAs on the average number of with-in range solutions as verification in Table 3. Since SOGAs on one metric perform poorly (≤ 1), we only display MOEAs on three metrics individually and SOGAs on weighted sum of three metrics. When $K = 5, 20, 30$, MOEA/D has the highest # with-in range solutions, while SPEA2 is the winner when $K = 50, 100$. This makes MOEA/D and SPEA2 the 1st and 2nd place in terms of ranking. NSGAI and PESAI are ranked as 3rd and 4th algorithms in most cases. GA, the best



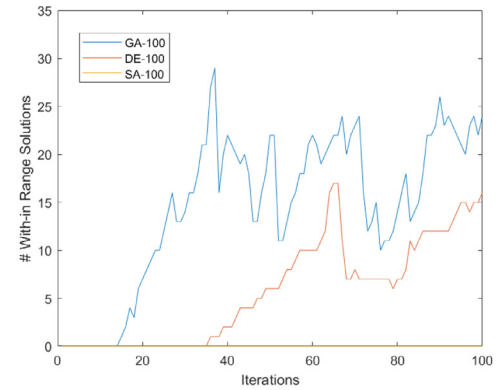
(a) Number of Pareto optimal solutions of SOGAs on one metric



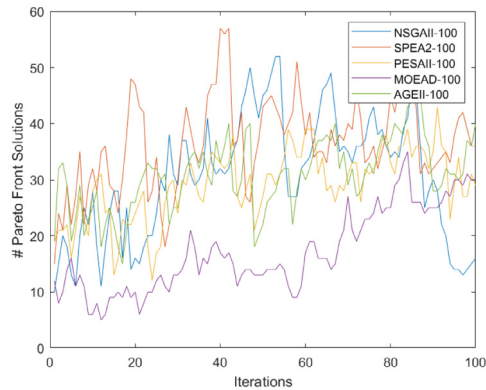
(b) Number of with-in range solutions of SOGAs on one metric



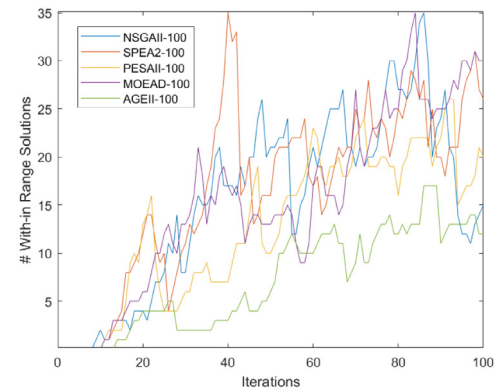
(c) Number of Pareto optimal solutions of SOGAs on weighted sum of three metrics



(d) Number of with-in range solutions of SOGAs on weighted sum of three metrics



(e) Number of Pareto optimal solutions of MOEAs on three metrics individually



(f) Number of with-in range solutions of MOEAs on three metrics individually

Fig. 7. Number of Pareto optimal solutions and with-in range solutions of SOGAs and MOEAs.

performing SOGA, is ranked in 3rd place for $K = 5$, 4th place for $K = 20, 100$ and in 5th place for $K = 30, 50$. This shows GA has fewer with-in range solutions in the middle of the run (at 50–90 iterations), and gets more solutions only in the last 5 iterations.

Result Discussion on Precision. Fig. 8 summarizes the percentage of with-in range solutions in Pareto set (precision) per iteration for the three sets of algorithms. The precision is 0% for SOGAs on one metric in (1) from Fig. 8(a). This is expected because SOGA with fitness function only optimizing one metric works poorly, as discussed earlier. From Fig. 8(b) and Fig. 8(c),

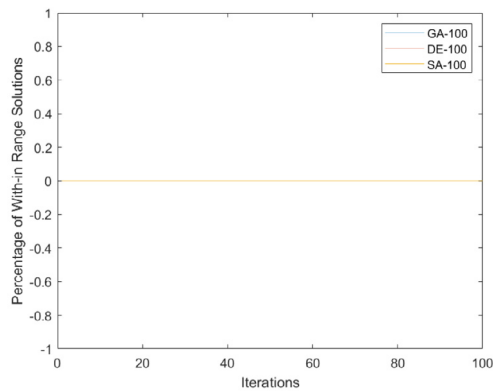
MOEA/D has much higher precision than the rest MOEAs and SOGAs in the early iterations (before iteration 50). GA is the top-performing SOGA in terms of precision. With a much smaller number of with-in range solutions being produced, GA climbed up to 100% precision at iteration 50. On the other hand, MOEA/D reached 100% precision at much earlier iteration (iteration 20), while having a larger number of with-in range solutions being produced.

Finally, we summarize the numeric results of MOEAs and SOGAs on the precision in Table 4. We used the same setting

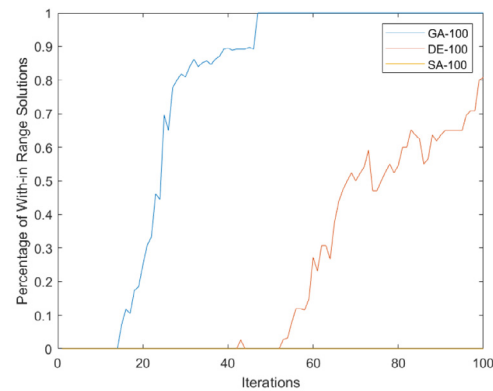
Table 4

Evaluation result on average percentage of with-in range solutions for MOEAs and SOGAs at the last iterations. Top 3 results in each column are highlighted in Bold.

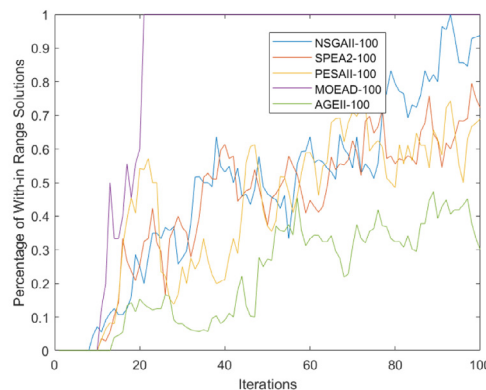
Algorithm	Category	% of With-in Range Solutions (Average of last K iterations)				
		$K = 5$	$K = 20$	$K = 30$	$K = 50$	$K = 100$
MOEA/D	MOEA	100%	100%	100%	100%	84%
AGEII	MOEA	38%	38%	37%	36%	22%
NSGAI	MOEA	90%	85%	78%	68%	48%
PESAII	MOEA	63%	62%	62%	59%	42%
SPEA2	MOEA	73%	65%	64%	58%	44%
GA	SOGA (Weighted Sum)	98%	98%	98%	97%	80%
DE	SOGA (Weighted Sum)	67%	61%	60%	45%	22%
SA	SOGA (Weighted Sum)	0%	0%	0%	0%	0%



(a) Number of Pareto optimal solutions of SOGAs on one metric



(b) Number of with-in range solutions of SOGAs on one metric



(c) Number of with-in range solutions of SOGAs on one metric

Fig. 8. Percentage of with-in range solutions in Pareto set of SOGAs on one metric, SOGAs on weighted sum of three metrics, and MOEAs.

as described in Table 3. The top 3 results in each column are highlighted in bold. With different K from 5, 20, 30, 50, 100, the 1st, 2nd and 3rd places are consistent: they are MOEA/D, GA and NSGAI. Compared with MOEA/D, although GA has a much smaller # with-in range solutions as shown in Table 3, it has the same level of precision as MOEA/D (close to 100% in the last 50 iterations). However, if we look at the average precision of all 100 iterations (when $K=100$), the precision of GA dropped to 80%, smaller than 84% for MOEA/D. This verifies that MOEA/D is able to reach high precision at earlier iterations than GA, which is an important advantage when the volume of users exposed in the A/B experiment and the time of running experiments are limited in the industrial setting. Comparing the precision result in Table 4 and the #with-in range solutions in Table 3, MOEA/D is ranked 1st or 2nd places consistently with different K for both precision and #with-in range solutions. On the other hand, although GA is

comparable with MOEA/D on precision, it is mostly ranked 4th, or 5th places on #with-in range solutions with varying K .

Summary for RQ2: MOEA/D and SPEA2, the two best-performing MOEAs, produce more high-quality (with-in range) solutions than SOGAs (SA, GA, DE), while MOEA/D, NSGAI, and GA have the highest precision.

5.4.3. RQ3: Effects of MOEA parameters

The input parameters of search-based techniques are important and it impacts the performance of the algorithms. Thus, we summarized the major parameters used in the algorithms (MOEAs and SOGAs) in Table 5. Although the outcome of a genetic algorithm is influenced by various parameters, such as mutation rate, crossover rate, elitism intensity, and population size, etc., the population size is a particularly hard parameter to determine for

Table 5
Parameter settings used in the search-based algorithms (MOEAs and SOGAs).

Parameters	Values
Population Size	100
Maximum Number of Iterations	100
Crossover Probability	100%
Mutation Probability	100%

a MOEA (Schaffer et al., 1989). Therefore, in this section, we focus on studying the effect of population sizes on the MOEA results in RQ3.

Fig. 9 shows the results of MOEA/D, one of the best-performing MOEAs in the experiment, with different population sizes. We keep the rest parameters the same and only change the population size in this experiment run. Results in the figure show that when population size is 20, 30 and 50, the MOEA/D converges slower than when population size is larger than 50. Besides that, different population sizes lead to achieving similar results in the final iteration, when the algorithm converges. From the result, there is no obvious change with different population sizes, except that the convergence is slower when population size is smaller than 50. For population size 150, 200 and 300, steep decreases are observed in less than 10 iterations. Thus, in cases where achieving convergence with small iterations is critical (e.g. due to limited user volume eligible for testing or long testing time to wait), a population size greater than 100 is recommended.

Summary for RQ3: Smaller population size affects the results only when the #iterations is limited at a small number. As the population size parameter decreases, the final performance stays the same, but the algorithm converges slower. When the iteration number becomes a constraint, larger population size for MOEA is recommended (e.g. use population size greater than 100 when the iteration number is limited at 10).

5.5. Discussion

From RQ1 and RQ3, we observe the same pattern of convergence regarding the results of MOEAs returned per iteration. The results are measured by either % change on three metrics or the distance between results and ground truth. As for the comparison between MOEAs and SOGAs, from RQ2, we validated that MOEAs with good performance (such as MOEA/D and SPEA2) have more high-quality solutions than SOGAs on the generated results. We summarize two main reasons for this:

- The candidate variant set from SOGA may include “useless” variants that are dominated by other variants, due to the single-objective nature of SOGA. These “useless” variants potentially limit the quality of candidate set.
- The candidate variant set from SOGA heavily depends on the method to select the “best” variants, while MOEA decouples the selection of the “best” variants, out of the candidate variant set generation. If the method to select variants is inaccurate or error-prone, the quality of candidate set would suffer. In fact, in A/B testing, the launch decision-making process of A/B test result (i.e., selecting variant to launch) is empirical and involves discussions and evaluations among domain experts (Xu et al., 2015; Tang et al., 2010), rather than simply using a weighted sum of objectives in SOGA. MOEA, on the other hand, does not have this issue.

We provide the connection back to an example to describe the implication of MOVSW in A/B testing. Let us go back to motivating example 1 in Fig. 1. In the current practice, engineers

(or designers) need to manually determine the values of the Font Size parameter in variants A, B and C, with the hope that these chosen values can lead to good A/B testing results. This manual process depends heavily on the skills of engineers, who typically have very little assistance or guidance on this. However, with the proposed MOVSW, the variants with new values of the Font Size parameter would be automatically created by MOEA and then evaluated in A/B test in each iteration of MOEA. Finally, MOVSW returns a candidate variant set as Pareto optimal set, which is passed to the decision makers to select the “best” variant to be launched. The benefits of MOVSW are mainly two-fold: (1) it frees the engineers from deciding and guessing which variants and parameter values to be used in the A/B test. (2) it automatically tries different combinations of parameter values in variants to obtain the optimal A/B results. In this example, the Font Size parameter as 14 with the best A/B test result would be captured by MOVSW, but might be missed if the parameter values of variants were chosen manually. If SOGA is used to automate this process, as proposed by Tamburrelli and Margara (2014), the numeric values of weights are needed by SOGA for metrics “Time spent”, “# user visits” and “# click through rate”, to produce a set of candidate variants.

5.6. Threats to validity

We discuss the potential threats to provide more guidance on the interpretation, limitation and other alternatives of the empirical experiment.

Construct validity. This threat relates to the potential bias of the news recommendation algorithm we select in the experiment. To mitigate this threat, although there are different alternative news recommendation algorithms available, we select the algorithm based on two criteria: (1) the algorithm should be as simple as possible, (2) the parameters should be sensitive enough to impact algorithm output. Based on these two criteria, we selected the simplified version of Decaying Model (Okura et al., 2017) as the algorithm.

Internal validity. This threat relates to the faulty conclusions caused by the limited MOEAs or SOGAs we evaluated. In this work, we selected and evaluated eight mainstream algorithms: five MOEAs and three SOGAs to reduce this threat. Our released code also enables other researchers to test on other MOEA or SOGA to extend on top of the evaluation results in this work.

External validity. This relates to the generality of the effectiveness of MOVSW on other A/B testing automation scenarios. To reduce this threat, in this work, we performed an extensive analysis of the results per iteration to demonstrate the convergence and effectiveness of MOVSW. However, we cannot make a sound claim on the effectiveness of MOVSW on another A/B testing scenario or dataset, given we have not tested on other scenarios. Another external validity relates to the implementation used in the case study. In the case study, we implemented the MOVSW in Matlab. For MOEA and SOGA, we directly used the library from PlatEMO (Tian et al., 2017). From the evaluation results, we believe that our implementation reflects the original methods.

Reliability. This threat is about the reproducibility of the results in the case study due to the nature of randomness in MOEA and SOGA. To mitigate this threat, we report the result of the case study extensively and release our code. This can allow other researchers to reproduce and extend our experiments in the case study.

6. Conclusions

In this paper, we formulated the Variant Creation and Evaluation (VCE) problem, and investigated the MOEA approach to

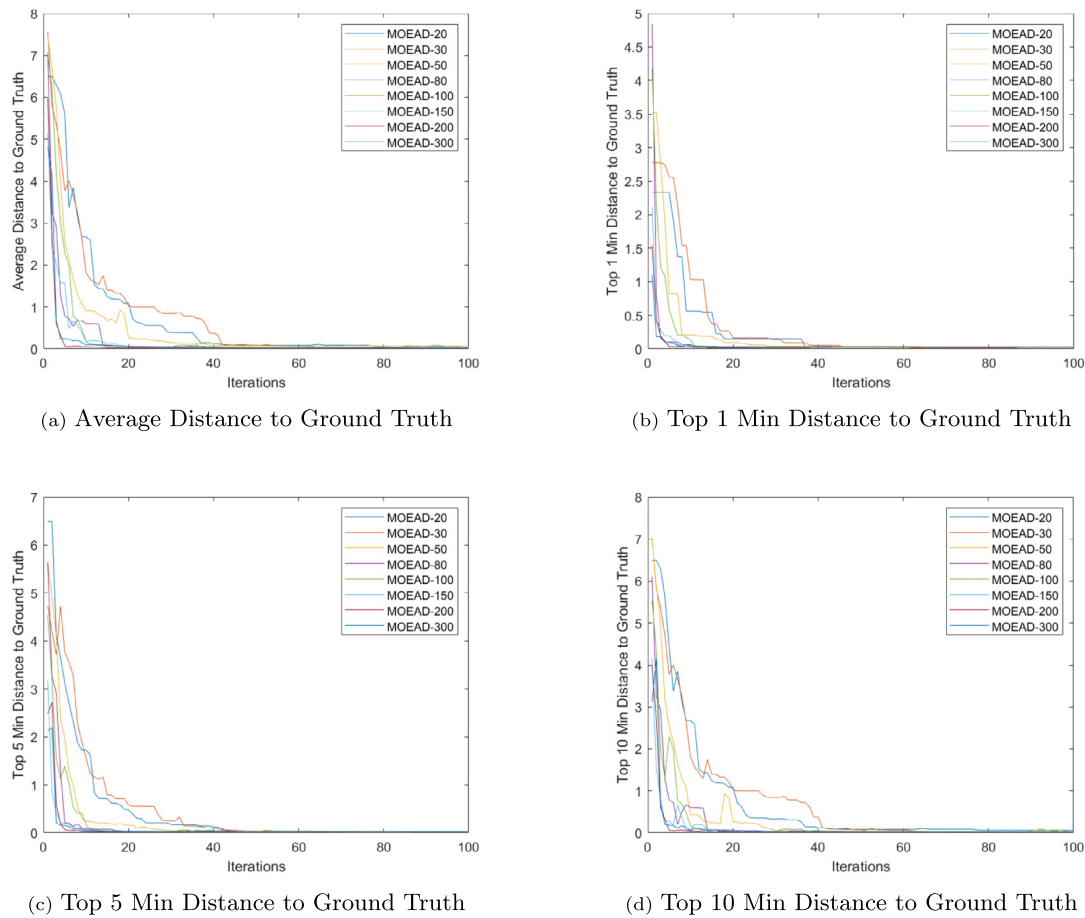


Fig. 9. Distance of Pareto Optimal Solutions to Ground Truth of MOEA/D algorithm with different population sizes.

selecting and evaluating variants toward automating the A/B test. While certain standards and empirical rules exist for creating variants in the A/B test, this process heavily relies upon human judgment, and is therefore manual, time-consuming and error-prone. We proposed the MOVSW method based on MOEA for addressing the VCE problem. In experiments, we conducted a case study to validate the effectiveness of applying MOEA on MOVSW to the VCE problem of optimizing parameter values in a news recommendation algorithm. Results show that among all the evaluated algorithms, MOEA/D achieves superior performance on the VCE problem, compared with other MOEA alternatives and single-objective genetic algorithms.

For future work, different types of solutions, either extending the MOEA approach or adopting other new approaches, to the VCE problem can be investigated and analyzed. Besides, one limitation of the MOVSW as the solution to the VCE problem is that it assumes that each variant created by MOEA can be exposed to any volume of users and can always get trustworthy metric results. However, the volume of users exposed to each variant in the A/B testing may be limited in industrial settings. Also, the time of running A/B experiments is a critical factor, since web-facing companies typically want the results as quickly as possible in a fast-paced environment. Thus, solutions that work under the constraints of exposed user volume in the tests and how long to run the tests can be explored.

CRediT authorship contribution statement

Jie J.W. Wu: Conceptualization, Methodology, Data curation, Writing – original draft, Writing – review & editing. **Thomas**

A. Mazzuchi: Supervision, Writing – review & editing. **Shahram Sarkani:** Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request

Acknowledgments

We would like to thank the anonymous reviewers for their constructive feedback and suggestions that greatly improved our paper and research.

References

- Adibi, M.A., Zandieh, Mostafa, Amiri, Maghsoud, 2010. Multi-objective scheduling of dynamic job shop using variable neighborhood search. *Expert Syst. Appl.* 37 (1), 282–287.
- Álvarez, Josefa Díaz, Risco-Martín, José L., Colmenar, J. Manuel, 2016. Multi-objective optimization of energy consumption and execution time in a single level cache memory for embedded systems. *J. Syst. Softw.* 111, 200–212.
- Anagnostopoulos, Konstantinos P., Mamanis, Georgios, 2011. The mean-variance cardinality constrained portfolio optimization problem: An experimental evaluation of five multiobjective evolutionary algorithms. *Expert Syst. Appl.* 38 (11), 14208–14217.

- Auer, Florian, Ros, Rasmus, Kaltenbrunner, Lukas, Runeson, Per, Felderer, Michael, 2021. Controlled experimentation in continuous experimentation: Knowledge and challenges. *Inf. Softw. Technol.* 134, 106551.
- Bertsimas, Dimitris, Tsitsiklis, John, 1993. Simulated annealing. *Statist. Sci.* 8 (1), 10–15.
- Bowman, Michael, Briand, Lionel C., Labiche, Yvan, 2010. Solving the class responsibility assignment problem in object-oriented analysis with multi-objective genetic algorithms. *IEEE Trans. Softw. Eng.* 36 (6), 817–837.
- Cai, Xingjuan, Hu, Zhaoming, Zhao, Peng, Zhang, Wensheng, Chen, Jinjun, 2020. A hybrid recommendation system with many-objective evolutionary algorithm. *Expert Syst. Appl.* 159, 113648.
- Clarke, John, Dolado, Jose Javier, Harman, Mark, Hierons, Rob, Jones, Bryan, Lumkin, Mary, Mitchell, Brian, Mancoridis, Spiros, Rees, Kearton, Roper, Marc, et al., 2003. Reformulating software engineering as a search problem. *IEE Proc.-Softw.* 150 (3), 161–175.
- Corne, David W., Jerram, Nick R., Knowles, Joshua D., Oates, Martin J., 2001. PESA-II: Region-based selection in evolutionary multiobjective optimization. In: *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation*. pp. 283–290.
- Davis, Lawrence, 1991. *Handbook of Genetic Algorithms*. CumInCAD.
- Deb, Kalyanmoy, Agrawal, Ram Bhushan, et al., 1995. Simulated binary crossover for continuous search space. *Complex Syst.* 9 (2), 115–148.
- Deb, Kalyanmoy, Pratap, Amrit, Agarwal, Sameer, Meyarivan, T.A.M.T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* 6 (2), 182–197.
- Deng, Alex, Shi, Xiaolin, 2016. Data-driven metric development for online controlled experiments: Seven lessons learned. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. pp. 77–86.
- Fabijan, Aleksander, Dmitriev, Pavel, Olsson, Helena Holmström, Bosch, Jan, 2017. The benefits of controlled experimentation at scale. In: *2017 43rd Euromicro Conference on Software Engineering and Advanced Applications*. SEAA, IEEE, pp. 18–26.
- Fabijan, Aleksander, Dmitriev, Pavel, Olsson, Helena Holmström, Bosch, Jan, 2018a. Effective online controlled experiment analysis at large scale. In: *2018 44th Euromicro Conference on Software Engineering and Advanced Applications*. SEAA, IEEE, pp. 64–67.
- Fabijan, Aleksander, Dmitriev, Pavel, Olsson, Helena Holmstrom, Bosch, Jan, 2018b. The online controlled experiment lifecycle. *IEEE Software* 37 (2), 60–67.
- Fagerholm, Fabian, Guinea, Alejandro Sanchez, Mäenpää, Hanna, Münch, Jürgen, 2017. The RIGHT model for continuous experimentation. *J. Syst. Softw.* 123, 292–305.
- Feitelson, Dror G., Frachtenberg, Eitan, Beck, Kent L., 2013. Development and deployment at facebook. *IEEE Internet Comput.* 17 (4), 8–17.
- Fitzgerald, Brian, Stol, Klaas-Jan, 2017. Continuous software engineering: A roadmap and agenda. *J. Syst. Softw.* 123, 176–189.
- Fonseca, Carlos M., Fleming, Peter J., 1998. Multiobjective optimization and multiple constraint handling with evolutionary algorithms. II. Application example. *IEEE Trans. Syst. Man Cybern. A* 28 (1), 38–47.
- George, E.P., Hunter, J. Stuart, Hunter, William Gordon, Bins, Roma, Kirlin IV, Kay, Carroll, Destiny, 2005. *Statistics for Experimenters: Design, Innovation, and Discovery*. Vol. 2, Wiley New York, NY, USA.
- Goldberg, David E., Holland, John Henry, 1988. Genetic algorithms and machine learning.
- Gomez-Urbe, Carlos A., Hunt, Neil, 2015. The netflix recommender system: Algorithms, business value, and innovation. *ACM Trans. Manag. Inf. Syst.* 6 (4), 1–19.
- Gulla, Jon Atle, Zhang, Lemei, Liu, Peng, Özgöbek, Özlem, Su, Xiaomeng, 2017. The adressa dataset for news recommendation. In: *Proceedings of the International Conference on Web Intelligence*. pp. 1042–1048.
- Guo, Jianmei, White, Jules, Wang, Guangxin, Li, Jian, Wang, Yinglin, 2011. A genetic algorithm for optimized feature selection with resource constraints in software product lines. *J. Syst. Softw.* 84 (12), 2208–2221.
- Hamamoto, Anderson Hiroshi, Carvalho, Luiz Fernando, Sampaio, Lucas Dias Hiera, Abrão, Taufik, Proença, Jr., Mario Lemes, 2018. Network anomaly detection system using genetic algorithm and fuzzy logic. *Expert Syst. Appl.* 92, 390–402.
- Hohnhold, Henning, O'Brien, Deirdre, Tang, Diane, 2015. Focusing on the long-term: It's good for users and business. In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. pp. 1849–1858.
- Holland, John Henry, et al., 1992. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. MIT Press.
- Kohavi, Ron, Deng, Alex, Frasca, Brian, Walker, Toby, Xu, Ya, Pohlmann, Nils, 2013. Online controlled experiments at large scale. In: *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. pp. 1168–1176.
- Kohavi, Ron, Henne, Randal M., Sommerfield, Dan, 2007. Practical guide to controlled experiments on the web: listen to your customers not to the hippo. In: *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. pp. 959–967.
- Kohavi, Ron, Longbotham, Roger, 2017. Online controlled experiments and A/B testing. *Encyclopedia Mach. Learn. Data Min.* 7 (8), 922–929.
- Kohavi, Ron, Tang, Diane, Xu, Ya, 2020. *Trustworthy Online Controlled Experiments: A Practical Guide to A/B Testing*. Cambridge University Press.
- Kohavi, Ron, Thomke, Stefan, 2017. The surprising power of online experiments. *Harv. Bus. Rev.* 95 (5), 74–82.
- Koning, Rembrand, Hasan, Sharique, Chatterji, Aaron, 2019. *Experimentation and Startup Performance: Evidence from A/B Testing*. Technical Report, National Bureau of Economic Research.
- Liao, Shu-Hsien, Hsieh, Chia-Lin, Lai, Peng-Jen, 2011. An evolutionary approach for multi-objective optimization of the integrated location-inventory distribution network problem in vendor-managed inventory. *Expert Syst. Appl.* 38 (6), 6768–6776.
- Macedo, Luís Lobato, Godinho, Pedro, Alves, Maria João, 2017. Mean-semivariance portfolio optimization with multiobjective evolutionary algorithms and technical analysis rules. *Expert Syst. Appl.* 79, 33–43.
- Machmouchi, Widad, Buscher, Georg, 2016. Principles for the design of online A/B metrics. In: *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*. pp. 589–590.
- Marcelino, CG, Leite, GMC, Delgado, CADM, de Oliveira, LB, Wanner, EF, Jiménez-Fernández, Silvia, Salcedo-Sanz, Sancho, 2021. An efficient multi-objective evolutionary approach for solving the operation of multi-reservoir system scheduling in hydro-power plants. *Expert Syst. Appl.* 185, 115638.
- Mirjalili, Seyedali, Saremi, Shahrzad, Mirjalili, Seyed Mohammad, Coelho, Leandro dos S., 2016. Multi-objective grey wolf optimizer: a novel algorithm for multi-criterion optimization. *Expert Syst. Appl.* 47, 106–119.
- Mukhopadhyay, Anirban, Maulik, Ujjwal, Bandyopadhyay, Sanghamitra, Coello, Carlos Artemio Coello, 2013. A survey of multiobjective evolutionary algorithms for data mining: Part I. *IEEE Trans. Evol. Comput.* 18 (1), 4–19.
- Murata, Tadahiko, Ishibuchi, Hisao, Tanaka, Hideo, 1996. Multi-objective genetic algorithm and its applications to flowshop scheduling. *Comput. Ind. Eng.* 30 (4), 957–968.
- Nazarahari, Milad, Khanmirza, Esmael, Doostie, Samira, 2019. Multi-objective multi-robot path planning in continuous environment using an enhanced genetic algorithm. *Expert Syst. Appl.* 115, 106–120.
- Ni, Chao, Chen, Xiang, Wu, Fangfang, Shen, Yuxiang, Gu, Qing, 2019. An empirical study on pareto based multi-objective feature selection for software defect prediction. *J. Syst. Softw.* 152, 215–238.
- Okura, Shumpei, Tagami, Yukihiro, Ono, Shingo, Tajima, Akira, 2017. Embedding-based news recommendation for millions of users. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. pp. 1933–1942.
- Olsson, Helena Holmström, 2018. Challenges and strategies for undertaking continuous experimentation to embedded systems: Industry and research perspectives. *Agile Process. Softw. Eng. Extreme Program.* 277.
- Oreski, Stjepan, Oreski, Goran, 2014. Genetic algorithm-based heuristic for feature selection in credit risk assessment. *Expert Syst. Appl.* 41 (4), 2052–2064.
- Pajkovic, Niko, 2022. Algorithms and taste-making: Exposing the Netflix Recommender System's operational logics. *Convergence* 28 (1), 214–235.
- Parejo, José A., Sánchez, Ana B., Segura, Sergio, Ruiz-Cortés, Antonio, Lopez-Herrejon, Roberto E., Egyed, Alexander, 2016. Multi-objective test case prioritization in highly configurable systems: A case study. *J. Syst. Softw.* 122, 287–310.
- Pascual, Gustavo G., Lopez-Herrejon, Roberto E, Pinto, Mónica, Fuentes, Lidia, Egyed, Alexander, 2015. Applying multiobjective evolutionary algorithms to dynamic software product lines for reconfiguring mobile applications. *J. Syst. Softw.* 103, 392–411.
- Pradhan, Dipesh, Wang, Shuai, Ali, Shaukat, Yue, Tao, Liaaen, Marius, 2019. Employing rule mining and multi-objective search for dynamic test case prioritization. *J. Syst. Softw.* 153, 86–104.
- Ramirez, Aurora, Romero, José Raúl, Ventura, Sebastian, 2019. A survey of many-objective optimisation in search-based software engineering. *J. Syst. Softw.* 149, 382–395.
- Reddy, M. Janga, Kumar, D. Nagesh, 2006. Optimal reservoir operation using multi-objective evolutionary algorithm. *Water Resour. Manag.* 20 (6), 861–878.
- Schaffer, J. David, Caruana, Rich, Eshelman, Larry J., Das, Rajarshi, 1989. A study of control parameters affecting online performance of genetic algorithms for function optimization. In: *Proceedings of the 3rd International Conference on Genetic Algorithms*. pp. 51–60.
- Storn, Rainer, Price, Kenneth, 1997. Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *J. Global Optim.* 11 (4), 341–359, <https://link.springer.com/article/10.1023/a:1008202821328>.

- Tamburrelli, Giordano, Margara, Alessandro, 2014. Towards automated a/b testing. In: *International Symposium on Search Based Software Engineering*. Springer, pp. 184–198.
- Tang, Diane, Agarwal, Ashish, O'Brien, Deirdre, Meyer, Mike, 2010. Overlapping experiment infrastructure: More, better, faster experimentation. In: *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. pp. 17–26, <https://dl.acm.org/doi/10.1145/1835804.1835810>.
- Tavana, Madjid, Abtahi, Amir-Reza, Khalili-Damghani, Kaveh, 2014. A new multi-objective multi-mode model for solving preemptive time–cost–quality trade-off project scheduling problems. *Expert Syst. Appl.* 41 (4), 1830–1846.
- Tian, Ye, Cheng, Ran, Zhang, Xingyi, Jin, Yaochu, 2017. PlatEMO: A MATLAB platform for evolutionary multi-objective optimization [educational forum]. *IEEE Comput. Intell. Mag.* 12 (4), 73–87.
- Turnbull, Bradley C., 2019. Learning intent to book metrics for airbnb search. In: *The World Wide Web Conference*. pp. 3265–3271.
- Van Veldhuizen, David A., Lamont, Gary B., 1998. Multiobjective Evolutionary Algorithm Research: A History and Analysis. Technical Report, Citeseer.
- Van Veldhuizen, David A., Lamont, Gary B., 2000. Multiobjective evolutionary algorithms: Analyzing the state-of-the-art. *Evol. Comput.* 8 (2), 125–147.
- Vignolo, Leandro D., Milone, Diego H., Scharcanski, Jacob, 2013. Feature selection for face recognition based on multi-objective evolutionary wrappers. *Expert Syst. Appl.* 40 (13), 5077–5084.
- Wagner, Markus, Neumann, Frank, 2013. A fast approximation-guided evolutionary multi-objective algorithm. In: *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*. pp. 687–694, <https://dl.acm.org/doi/10.1145/2463372.2463448>.
- Wang, Tianshu, Zhang, Gongxuan, Yang, Xichen, Vajdi, Ahmadreza, 2018. Genetic algorithm for energy-efficient clustering and routing in wireless sensor networks. *J. Syst. Softw.* 146, 196–214.
- Wu, Jie J.W., Mazzuchi, Thomas A., Sarkani, Shahram, 2023. Comparison of multi-criteria decision-making methods for online controlled experiments in a launch decision-making framework. *Inf. Softw. Technol.* 155, 107115.
- Xu, Ya, Chen, Nanyu, Fernandez, Addrian, Sinno, Omar, Bhasin, Anmol, 2015. From infrastructure to culture: A/B testing challenges in large scale social networks. In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. pp. 2227–2236, <https://dl.acm.org/doi/10.1145/2783258.2788602>.
- Yeh, Wei-Chang, Chuang, Mei-Chi, 2011. Using multi-objective genetic algorithm for partner selection in green supply chain problems. *Expert Syst. Appl.* 38 (4), 4244–4253.
- Zhang, Qingfu, Li, Hui, 2007. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Trans. Evol. Comput.* 11 (6), 712–731.
- Zitzler, Eckart, Laumanns, Marco, Thiele, Lothar, 2001. SPEA2: Improving the strength Pareto evolutionary algorithm. *TIK-Rep.* 103.



Jie J.W. Wu received the B.S. and M.S. degree in Computer Science from Shanghai Jiao Tong University, China, in 2012 and 2015, respectively. Since 2015, he has been in the software industry in the U.S., and he is currently a software engineer at Snap Inc. He is currently pursuing the Ph.D. degree in system engineering with George Washington University, Washington, DC, USA. He is mainly interested in automated techniques and system thinking in software engineering.



Thomas A. Mazzuchi received the B.A. degree in mathematics from Gettysburg College, Gettysburg, PA, USA, in 1978, and the M.S. and D.Sc. degrees in operations research, from The George Washington University (GW), Washington, D.C., USA, in 1979 and 1982, respectively.

He is a Professor of engineering management and systems engineering, and Chair of the Department of Engineering Management and Systems Engineering, in the School of Engineering and Applied Science, GW. Formerly, he served as the Chair of the Department of Operations Research, and as Interim Dean of the School of Engineering and Applied Science. He has been engaged in consulting and research in the areas of reliability and risk analysis, and systems engineering techniques, for over 30 years. He served for 2.5 years as a Research Mathematician at the International Operations and Process Research Laboratory of the Royal Dutch Shell Company. While at Shell, he was engaged in reliability and risk analysis of large processing systems, maintenance optimization of off-shore platforms, and quality control procedures at large-scale chemical plants. In his academic career, he has held research contracts in development of testing procedures for both the U.S. Air Force and the U.S. Army; in spares provisioning modeling with the U.S. Postal Service; in mission assurance with NASA; and in maritime safety and risk assessment with the Port Authority of New Orleans, the Washington Office of Marine Safety, the Washington State Department of Transportation, and the San Francisco Bay Area Transit Authority.

Dr. Mazzuchi is an Elected Member of the International Statistics Institute.



Shahram Sarkani received the B.S. and M.S. degrees in civil engineering from Louisiana State University, Baton Rouge, LA, USA, in 1980 and 1981, respectively, and the Ph.D. degree in civil engineering from Rice University, Houston, TX, USA, in 1987.

He is a Professor of engineering management and systems engineering with The George Washington University (GW), Washington, D.C., USA. His current administrative appointments are Inaugural Director, School of Engineering and Applied Science off-campus and Professional Programs since 2016, the school unit to establish cross-disciplinary and departmental programs for offer off-campus and/or by synchronous distance learning; and Faculty Adviser and Academic Director, EMSEoff-campus Programs since 2001, the department unit that designs and administers five separate graduate degree programs in six areas of study that enroll over 800 students across the USA and abroad. He joined GW in 1986, where previous administrative appointments include Chair of the Civil, Mechanical, and Environmental Engineering Department (1994–1997); and Interim Associate Dean for Research, School of Engineering and Applied Science (1997–2001). In over 500 technical publications and presentations, his research in systems engineering, systems analysis, and applied enterprise systems engineering has application to risk analysis, structural safety, and reliability. He has conducted sponsored research for such organizations as NASA, NIST, NSF, U.S. AID, and the U.S. Departments of Interior, Navy, and Transportation.

Dr. Sarkani received the Walter L. Huber Civil Engineering Research Prize by the American Society of Civil Engineers in 1999. He was inducted into the Civil and Environmental Engineering Hall of Distinction, Louisiana State University, in 2010. He is a Registered Professional Engineer in Virginia.