Uncovering Hazards Using Multi-Objective Optimization to Explore the Faulty State-Space

Inga A. Girshfeld *
USRA (NASA Ames Research Center), Moffett Field, CA, 95035
University of Southern California, Los Angeles, CA 90007

Daniel Hulse†
NASA Ames Research Center, Moffett Field, CA, 95035

Lukman Irshad‡
KBR, Inc (NASA Ames Research Center), Moffett Field, CA, 95035

Abstract

A key part of resilience analysis is identifying modes of failure which may affect system function. While there is an opportunity for scenario generation approaches to augment the tedious and error-prone process of mode identification, existing approaches are not suited to this task, often returning sets of scenarios which are essentially redundant with each other, rather than a set of distinct modes. In this work, we demonstrate an approach to enable computational failure mode identification by generating a tractable, unique set of hazardous parameters using multi-objective optimization. In this approach, a set of scenarios is optimized along two objectives characterizing scenario uniqueness and severity. To best solve this problem, we propose the use of a cooperative co-evolutionary algorithm (CCEA) that separates the generation and optimization of individual scenarios from the selection of the combined set. To understand the performance and usefulness of this approach, we demonstrate it in the generation of drive fault scenarios in an autonomous rover case study. We then use this case study to (1) benchmark the CCEA algorithm against a more conventional evolutionary algorithm and (2) explore the solution trade-space over input and response-based uniqueness objectives to better understand the effect of objective formulation on solution quality. As shown, this approach enables the discovery of a variety of heterogeneous high-severity scenarios, however, its performance at this task is highly sensitive to the formulated objectives and their weights.

I. Introduction

One major consideration in the design of resilience is the mitigation of unforeseen "black swan" hazardous scenarios [1, 2]. While there a variety of methods to achieve this, such as developing generic properties of fault tolerance and "graceful extensibility," one of the most effective means is to try to better identify and thus foresee and account for these scenarios directly [3, 4]. While this is often thought of as a discursive process (i.e., where designers think about possible failure paths [5]), there is opportunity to leverage computational scenario generation methods to explore the system state-space to identify novel scenarios which a design team would not have thought of otherwise.

Previous work using scenario generation for resilience-based design has largely focused on the design of infrastructure. One major application is the management of renewable power sources (such as wind [6], solar, and hydro [7]) for forecasting potential future power output and loads [8] in a future energy environment [9] given current and past operational data [10, 11]. This can in turn inform operators or even enable the optimization (see: [11]) of energy allocation in response to these potentialities. Scenario generation is also used in the resilience-based design of infrastructure systems to simulate hazards which may occur in operations. Some examples include the generation of natural disasters (e.g., winter storms [12] and wildfires [13]) to model their effect on power distribution [14], the synthetic generation of droughts to model their effect on urban water distribution systems [15], and the generation of meteorological conditions to quantify their effect on airport pavement networks [16]. While the goal of these methods is

*Research Intern, Intelligent Systems Division, Mail Stop N269-2, Moffett Field, CA 94035
†Software Systems AST, Intelligent Systems Division, Mail Stop N269-2, Moffett Field, CA 94035
‡Research Engineer, Intelligent Systems Division, Mail Stop N269-2, Moffett Field, CA 94035

generally to generate "plausible" scenarios based on historical data, exploratory analysis outside the sample region can additionally help identify potential scenarios when there is no data (e.g., in a novel system design) [15].

The use of scenario generation for the general design of resilience is a relatively new concept. Previous work incorporating scenario generation either focused on specific applications (e.g., cognition [17], infrastructure, etc.) or on the ability to generate increasingly complex types of scenarios for evaluation (i.e., joint failures [18–20]). Some existing research on algorithmic scenario identification has additionally been developed to identify risk scenarios from structural attributes [21] and search for most likely failure scenario given a probabilistic model [22]. To explore how scenario generation could be used in resilience-based design to discover unforeseen hazards, the authors previously developed the technique of synthetic mode generation [23]. This method extended the traditional mode identification process by sampling a space of potential hazardous states, which was shown to increase the number of discrete failure trajectories revealed compared to manual identification. However, this approach came at a significant amount of computational expense and generated many essentially duplicate modes because it was based on an exhaustive search. This problem is additionally present in many search-based scenario generation approaches, which typically converge on a worst-case failure scenario (and scenarios similar to it) unless guided by domain knowledge [24], thus providing limited and redundant information about potential failure paths.

A. Contribution

To ensure the many unique ways a system will fail are mitigated in design, it is instead desirable for scenario generation methods to not just return the set of worst-case scenarios (which may be redundant with each other each other), but a set of heterogeneous failure modalities. To enable this, previous work posed the scenario generation task as a multi-objective optimization problem with objectives for (1) the aggregate hazard function (e.g. severity, probability, risk, etc.) and (2) the aggregate uniqueness of the modes, defined as the nearest neighbor distance of the states [25]. A diversity-based genetic algorithm was further proposed to solve this type of problem, with an early proof of concept demonstration returning good results from the approach. In this paper, we intend to further study (1) the multi-objective nature of this formulation of the scenario generation problem (i.e., the trade-off between aggregate severity and uniqueness) and (2) the effectiveness of a Cooperative Co-evolutionary algorithm on this domain, which we hypothesize will perform well in this domain by separating the optimal generation of each mode from the optimal combination of modes in a set. This study will be performed over an autonomous rover case study in Section IV, where the algorithm is used go generate modes in the drive system. The remaining sections provide background for the methodology (Section II), outline the approach (Section III), and discuss the findings of this case study demonstration (Section V-VI.

II. Background

To further contextualize this work, this section presents background on previous Scenario Generation approaches (Section II.A) as well as the Cooperative Coevolutionary Algorithm (Section II.B) adapted in this work to the mode generation problem.

A. Scenario Generation

The concept of scenario generation has been formally defined in the field of stochastic programming as a set of approaches used to accurately represent an underlying probability distribution with a few defined points, or scenarios [26–28]. A wide variety of methods have been formulated for this, including monte carlo (and pseudo-monte carlo) methods [28], hidden markov models [29], moment matching [30], finite-element/discretization [31], tree search, and numerical integration-type approaches [26]. This formulation has application to stochastic programming problems in which the goal is to make an optimal choices that will result in uncertain outcomes (e.g., design under uncertainty or portfolio optimization problems), where it has been shown that different risk attitudes can effect the choice of generation strategy [32]. However, considering scenario generation techniques solely within the context of stochastic programming techniques limit their usefulness–specifically, when the underlying probability distributions are not known or there is not good real-world data [27], making this approach difficult to rely on for every decision. Scenario generation has thus come to encompass a wide range of techniques for identifying particular sequences of events with a wide range of applications outside of optimal decision-making under uncertainty.

Scenario generation is used in many human-in-the-loop simulation approaches where it is desirable to keep the number of scenarios small due to the costs of working with human participants. Scenario generation is thus often used

to develop training simulations for first responders [33, 34] and military personnel [35, 36] to ensure that they learn how to operate systems in a wide range of situations. While many of the underlying techniques for training scenario generation share similarities with traditional approaches, they have been heavily modified toward meeting the individual needs of the students [35], rather than optimizing decisions. Outside of the training application, scenario generation has been used in human-in-the-loop simulations for simulation of air traffic control systems [37, 38] and exploring the space of potential product use-cases [39] to limit the number of simulations each participant needs to perform over while still presenting a variety of circumstances.

Scenario generation approaches are used often in the design and testing of software and systems, where the goal is to generate test cases over which the software can be evaluated to have correct (or incorrect) behavior. In these approaches, the goal is generally to achieve product safety/quality by ensuring that the designed system (1) performs as desired in a wide range of use-cases without introducing undesired failures and/or (2) effectively mitigates potentially hazardous scenarios. In the broader field of systems engineering, scenario generation has been used in the context of a system model to ensure that the structure and operations actively mitigate risks [21, 40, 41]. In software engineering, scenario generation approaches have been traditionally used to ensure a given program will function correctly over every single edge-case [42–44] based on an interaction model. Newer, more challenging software applications (e.g., automated driving) have vastly increased the need for sophisticated scenario generation approaches, because the space of potential situations is very large and the software has an active role in maintaining system safety.

In the design of autonomous vehicles specifically, scenario generation procedures are being developed to test that a vehicle can respond to the wide range of simulated driving situations normally expected of a driver [45]. Three types of approaches have been presented for this–data driven approaches (where previous data is used to sample or generate realistic driving scenarios), knowledge-based generation (where known cases are used directly or used to guide a scenario generator), and adversarial generation (where the goal is to find scenarios most likely to bring the system to failure) [46]. Adversarial generation approaches are split into methods that generate static scenarios (single-event scenarios where parameters are set at a single timestep, e.g., there is an object in the roadway) and dynamic scenarios (multi-event scenarios with different parameters at each timestep) [46]. Stress testing is a type of dynamic scenario approach which formulates the generation of likely failure scenarios as a markov decision process [47], and has been used both for automated driving [48] and aircraft collision avoidance [49, 50]. One of the difficulties with stress testing is that it often returns many very similar failure scenarios, rather than all of the potential paths to failure, a problem which has previously been solved by augmenting the search with domain knowledge [24]. A previous static generation approach has proposed a diversity-based genetic algorithm to better explore the space of potential hazards in scenario generation [25]. In this research, we seek to advance on this approach by (1) further defining and exploring the multi objective nature of the problem and (2) presenting a cooperative co-evolutionary algorithm adapted to this type of problem.

B. Cooperative Co-evolutionary Algorithm

Cooperative co-evolutionary algorithms (CCEA) were developed to solve optimization problems of increased complexity, specifically, with interacting co-adaptive sub-components [51]. Inspired by evolutionary mutualism, an interaction within biological systems between species that is beneficial to both [52], CCEAs take into consideration the effects of interactions between species (parts or components of the overall system) in the evolution of the overall solution vector. This is in contrast to traditional evolutionary algorithms which evolve the entire solution vector monotonically [53]. This can be helpful in optimization problems where coupling relationships between components require them to be co-optimized together [54]. To implement a CCEA, the complex optimization problem is decomposed to "species" sub-populations which correspond to the partitions of the overall solution vector, which may be permuted at each iteration. Then, individual representatives from each sub-population are selected, their fitness is evaluated with respect to the other representatives, and selected for the next generation [51, 53].

In systems with readily-decomposable components (e.g., multiagent systems [55]) with "local" and "global" objectives to be considered at each level, CCEAs have been observed to enable increased optimization performance by accounting for the problem structure [56]. These decentralized optimization domains (e.g., [57]) are analogous to the problem considered in this paper, where individual mode properties (i.e., severity) can be readily decomposed while others (i.e., uniqueness) require evaluation in the context of a set. We thus hypothesize that a well-implemented CCEA can outperform a similarly-constructed evolutionary algorithm on the scenario generation problem.

III. Methodology

This paper presents a method to search for unique hazardous modes in the faulty state-space, a vector space which defines input to the system model which includes (but is not limited by) known failure scenario attributes. To present this method, Section III.A shows the multi-objective optimization-based formulation of the scenario generation problem, which, when solved, returns a set of hazardous and unique failure modes. To most effectively solve this problem, Section III.B then proposes a cooperative coevolutionary algorithm for searching modes with the state-space and selecting the best (most hazardous and heterogeneous) set of modes.

A. Problem Definition

In this research, the task of finding a tractable set of unique modes (defined as perturbations of the system state) which cause the most harm is considered. This task may be formulated as the multi-objective problem of finding the set of modes $R = \{\vec{h}_i\}_{i=1}^{n}$ that maximizes the severity f_1 and uniqueness f_2 of the modes, where $\vec{h} = [h^1, \ldots, h^s]$ is the combination of s health states defining each of the n modes in the set. The resulting constrained multi-objective optimization problem is:

$$\begin{aligned} \underset{R}{\text{maximize}} \quad & f_1(R),\ f_2(R) \\ \text{subject to} \quad & g_j(R) \leq 0,\ j \in \mathbb{N} \\ & h_k(R) = 0,\ k \in \mathbb{N} \\ & R \in \Omega \subseteq \mathbb{R}^n \times \mathbb{R}^s. \end{aligned} \tag{1}$$

In general, the objective f_1 is a function representing the aggregate *severities* resulting from the simulation of each individual mode, which may be represented as a cost function C for each individual mode and evaluated over the set as the sum:

$$f_1(R) = \sum_{i=1}^{n} \mathrm{C}(\vec{h}_i) \tag{2}$$

We define the uniqueness objective f_2 in two different ways. In general, the uniqueness of the set of modes can be defined as the sum of the nearest-neighbor distances between the states of the model x:

$$f_2(R) = \sum_{j=1}^{n} \frac{\min_{i \neq j} ||\vec{x}_j - \vec{x}_i||_2}{N_j} \tag{3}$$

where N_j is an optional normalization factor for the dimension (e.g., $x_{max} - x_{min}$). Two formulations can thus be created with this form of objective in mind. In *Formulation 1*, the goal is to maximize the uniqueness of the modes in the faulty state-space, meaning that $\vec{x} = \vec{h}$. In *Formulation 2*, the goal is to maximize the uniqueness of the modes in the resulting set model states $\vec{e}$ resulting from the fault simulation (e.g., trajectories, damage, etc.). These two formulations will be explored and compared qualitatively in the Demonstration section (Section IV).

B. Algorithms

We propose solving the problem formalized in Equation 1 with a cooperative co-evolutionary algorithm (CCEA). CCEAs enable the combined optimization of individuals and teams in problems where individual and team performance can be separated (e.g., multiagent coordination problems). As shown in Figure 1, this can be contrasted with a more conventional evolutionary algorithm, where the entire team (or solution) is permuted and evaluated at once. We thus claim that CCEAs are well suited to the search of hazardous modes, where the goal is not just to provide high-impact modes (at the individual level), but also to provide a heterogeneous set (at the team level). While many implementations of cooperative coevolutionary algorithms exist [58], we develop our own implementation in this research which is adapted to the scenario search task defined in the problem defined in Equation 1. This algorithm is shown in Figure 1, with each step explained in detail in the next sub-sections.

1. Instantiation

During instantiation, each mode sub-population is initialized with random modes in the space. Mathematically, the algorithm defines *parents*, $P = \{S_j\}_{j=1}^{n}$, such that n is the number of *sub-populations* and $S_j = \{\vec{h}_i\}_{i=1}^{k}$, where k is the number of *individuals* per *sub-population* $\vec{h}_i = (h^1, \ldots, h^s)_i$, where s is the number of faulty states. At this point, each

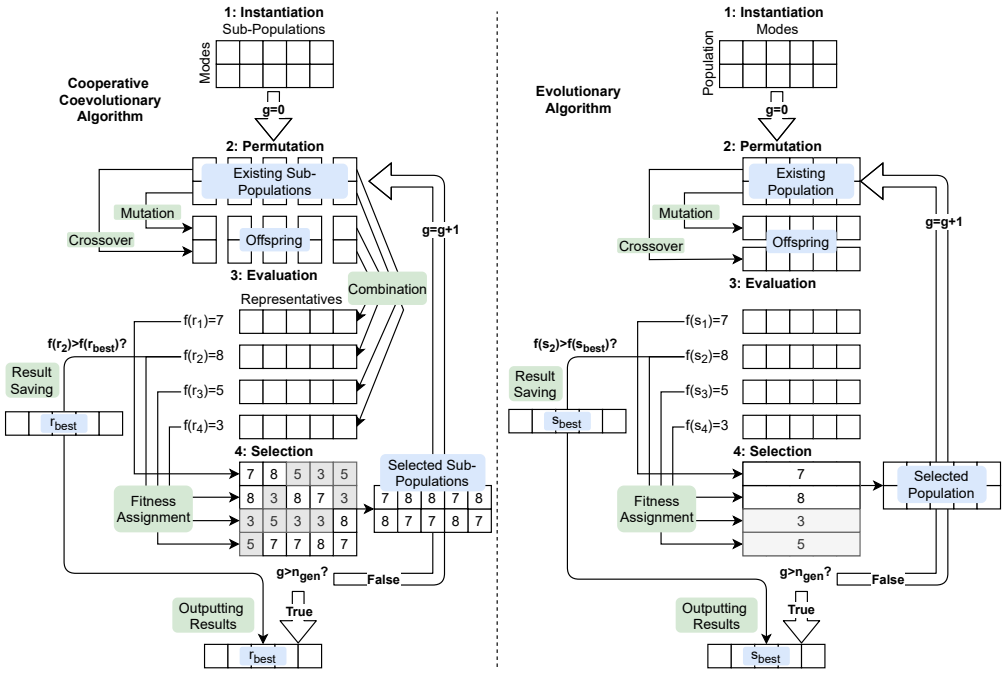


Fig. 1 Overall Cooperative Co-evolutionary Algorithm Compared to a Traditional Evolutionary Algorithm

of these modes are simulated in the model, giving the individual severities $C(h_j)$ and output states e_j which are saved as properties of the mode. Note these states are only evaluated once on instantiation to reduce the computational cost associated with re-evaluating already-simulated modes. This further enables simple evaluation of global fitness values in Step 3 and 4, which are relatively simple mathematical operations. Steps 2-4 are then performed in a loop:

2. Permutation

To supplement the existing sub-populations, a copy of the sub-population is permuted with mutation and crossover operations and added from a new overall set of sub-populations, resulting in the *offspring* population $O = \{S'_j\}_{j=1}^{n}$. In our implementation of the CCEA, the states of the mode $\vec{h}$ are permuted using a normal distribution in the mutation step:

$$h^i_{mut} = N(h^i_{old}, w * (h^i_{max} - h^i_{min}) \tag{4}$$

which is then truncated within the range of possible health-states:

$$h^i_{new} = \begin{cases} h^i_{min} & \text{if } h^i_{min} < h^i_{mut} \\ h^i_{max} & \text{if } h^i_{max} > h^i_{mut} \\ h^i_{mut} & \text{otherwise} \end{cases} \tag{5}$$

where h^i_{new} is the new faulty state value, h^i_{old} is the old faulty state value, w is a range factor (which was set to 0.25 after some experimentation) and h^i_{min} and h^i_{max} are the minimum and maximum faulty state values in the range.

Crossover, on the other hand, takes an existing mode and combines its states with the previous mode in the sub-population at a random point. As with the instantiation of new modes, these new modes are simulated as they are created to determine the resulting severities and output states to be used in the evaluation steps.

3. Evaluation

To evaluate the sub-populations, representative modes from each sub-population are combined to create representative solution vectors which can then be evaluated in terms of the objectives f_1 and f_2. In this process, each *sub-population* is sampled to generate a solution vector, $R_l = \{\vec{h}'_j\}_{j=1}^{n}$ for resulting in the set of *representatives* $\{R_l\}_{l=1}^{m}$. Since an

exhaustive elaboration of representatives is computationally costly ($O(n^k)$), m of these solution vectors are created at each iteration in this research, where m was set to 1000. It should be noted that many more sophisticated methods for creating this set exist in the literature, such as hall of fame evaluation [57], which may be explored in future work.

To evaluate the solutions, the objective values f_1 and f_2 are evaluated using the existing values stored in the modes created on instantiation. These objective values are then combined using the weighted sum approach:

$$f(R_l) = \frac{w}{C_{max} * n} * f_1(R_l) + \frac{1-w}{n\sqrt{s}} * f_2(R_l) \tag{6}$$

where C_{max} is the maximum severity cost, n is the number of modes, s is the number of dimensions making up the state-space defining mode uniqueness, and w is a weighting factor chosen by the analyst to weight objectives. At this point, the vector R_l is compared with the previous best representative solution and replaced if it has a higher overall objective value. The algorithm may then terminate if the maximum number of generations is reached.

4. Selection

Finally, the modes are selected based on their performance in the representatives. In this work, we use leniency (see: [57]) as a heuristic to assign credit to modes, meaning that the local fitness for each mode in each sub-population is determined to be the maximum value of the representatives it is involved in, or

$$f(h^i) = \max_{R \; s.t. \; h^i \in R} (f(R)) \tag{7}$$

Based on these fitness values, each sub-population undergoes a selection process, in this case taking the modes with the top 50% of local fitness values to carry on to the next generation.

IV. Demonstration

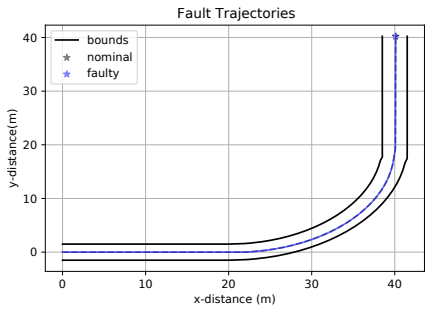
To demonstrate the use of the multi-objective optimization formulation of the scenario generation problem as well as the performance of the CCEA algorithm, we consider a case study of finding fault modes in the drive system of an autonomous rover. This rover was developed in previously to study a variety of resilience simulation methodologies, including the exhaustive generation of failure modes [23]. The goal of the rover is to perform a basic autonomous driving task by following a line-marking from a given start to given end location. The structure of the rover can be viewed in Figure 2b and is comprised of control, avionics, drive and power systems, in addition to its interaction with the environment. While there are a variety of possibilities for input lines for the rover to follow, the route used in this paper for demonstration purposes follows an L-Curve as seen in Figure 2a. If the rover deviates from the center line, it may go off course and crash into its surroundings. If it deviates more than a meter from the center line, the rover can no longer see the center-line and stops moving because the rover has crashed.

Modes considered in this case study a mode are made up of the faulty state vector $\vec{h} = [h^f, h^d, h^t]$ where h^f, h^d, and h^t represents friction, drift and transfer in the drive system, respectively. The severity of these modes is evaluated by injecting them as the rover enters the turn and simulating the progression of the rover until it stops itself, crashes, or reaches the desired end-location, at which the end-location and deviation from the center-line is recorded. This line distance measure $d_l(\vec{h}_j)$ is used in this work as the *severity* of fault modes in f_1. Thus:

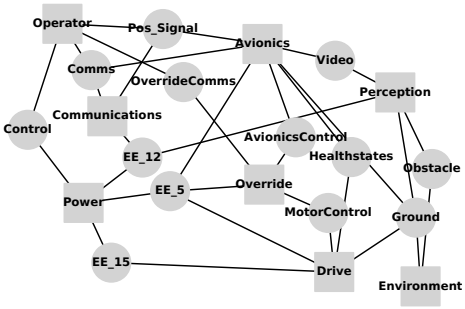
$$\mathrm{C}(\vec{h_j}) = d_l(\vec{h}_j). \tag{8}$$

The set of modes to identify is then $R = \{\vec{h}_j\}_{j=1}^{n}$, $1 \leq l \leq n^k$, a set of n desired $\vec{h}_j$ vectors that maximizes this line distance metric d_l in Equation 2 as well as the uniqueness metric in Equation 3.

In the next subsections, two formulations of the multi-objective search problem are used for this rover. In the first formulation, objective for the uniqueness of solutions (Equation 3) is defined in terms of distance in the faulty state-space (i.e., distance between fault-state vectors). In the second formulation, solution uniqueness is defined in terms of distance in the results space (i.e., distance between rover final end-locations). In the next subsections, each of these formulations are studied and compared qualitatively on the basis of both uniqueness considerations (spanning the faulty state-space and results-space), as well as the resulting distribution. To form the basis for this comparison, the CCEA algorithm is first bench-marked to show its overall effectiveness.



(a) Chosen L-shaped rover trajectory



(b) Structure of rover model in this case study.

Fig. 2 Rover Case Study Model Environment and Structure

A. Algorithm Performance Comparison

To test the merit of the CCEA algorithm on the mode search problem, this section compares the CCEA algorithm with an evolutionary algorithm and monte carlo search. To perform this comparison, the algorithms were run using similar parameters over 100 generations in Formulation 1 of the rover optimization problems with equally-weighted objectives. The resulting average best solution over 20 runs is shown in Figure 3. As shown, the CCEA consistently outperforms the conventional evolutionary algorithm, especially early in the search, meaning that it is both a more efficient and effective optimization strategy. Both evolutionary algorithms also consistently outperform the monte carlo search strategy, which seemingly approaches an asymptote. This makes sense because both evolutionary searches have the ability to reuse previous high-value points and their evaluations at each generation, while each generation of the monte carlo search is essentially a random solution, making the search slower (since all individuals must be simulated at each generation) and less targeted (since good individuals cannot be reused/permuted).

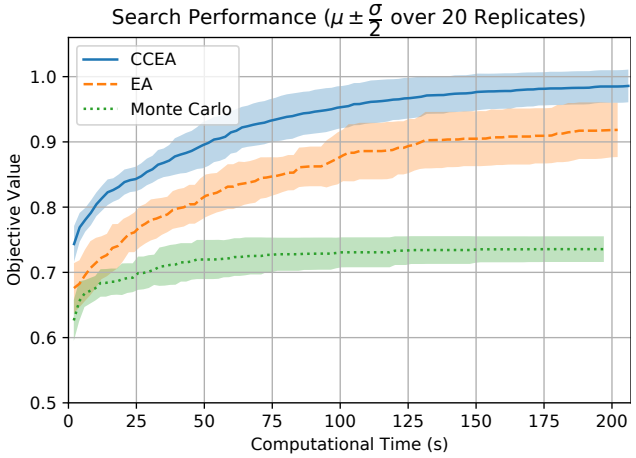


Fig. 3 Comparison of CCEA algorithm with EA and Monte Carlo algorithms for joint faulty state-space exploration

Additionally, the performance of the CCEA can be attributed to the more effective permutation and combination of individuals. In the EA, mutation and crossover happen on the mode vectors that make up a given solution in the population. In the CCEA, mutation and crossover happen within sub-populations which are then combined to make an overall solution. This separation enables each mode to be varied (rather than an entire solution) individually and then combined (with both mutations and previous good solutions in the other sub-populations) to form the best overall solution. The CCEA approach can also result in better simulation efficiencies per generation compared to the EA, since each new solution does not require a re-simulation of the constituent modes (as opposed to the EA permutation/evaluation step, which occurs for the entire solution), and evaluating these solutions requires a relatively inexpensive tabulation of the objectives from the results saved for each mode. While this shows the merit of the CCEA approach, there may be

other formulations of the EA which could improve performance, such as modifying the permutation mutation/crossover functions to better match the problem, for example by swapping points instead of state-values or permuting a narrower set of points. Similarly, the CCEA approach could additionally be improved by enabling the sharing of individuals between sub-populations and using a better-performing combination method (e.g., those in Ref. [57]). However, this demonstration shows the general merit of the CCEA approach as an effective search method for generating unique failure modes.

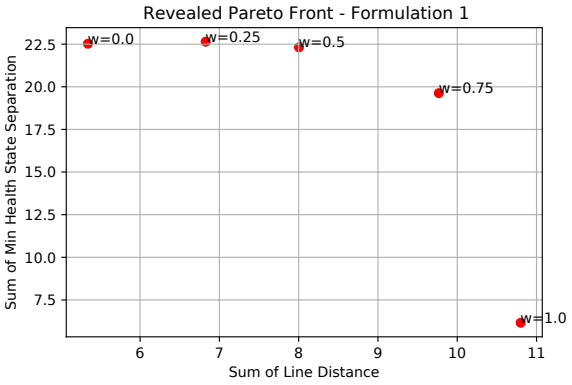


Fig. 4 Found Pareto front of Line Distance and Faulty State-Space Distance in Formulation 1

B. Formulation 1: State-Space Distance

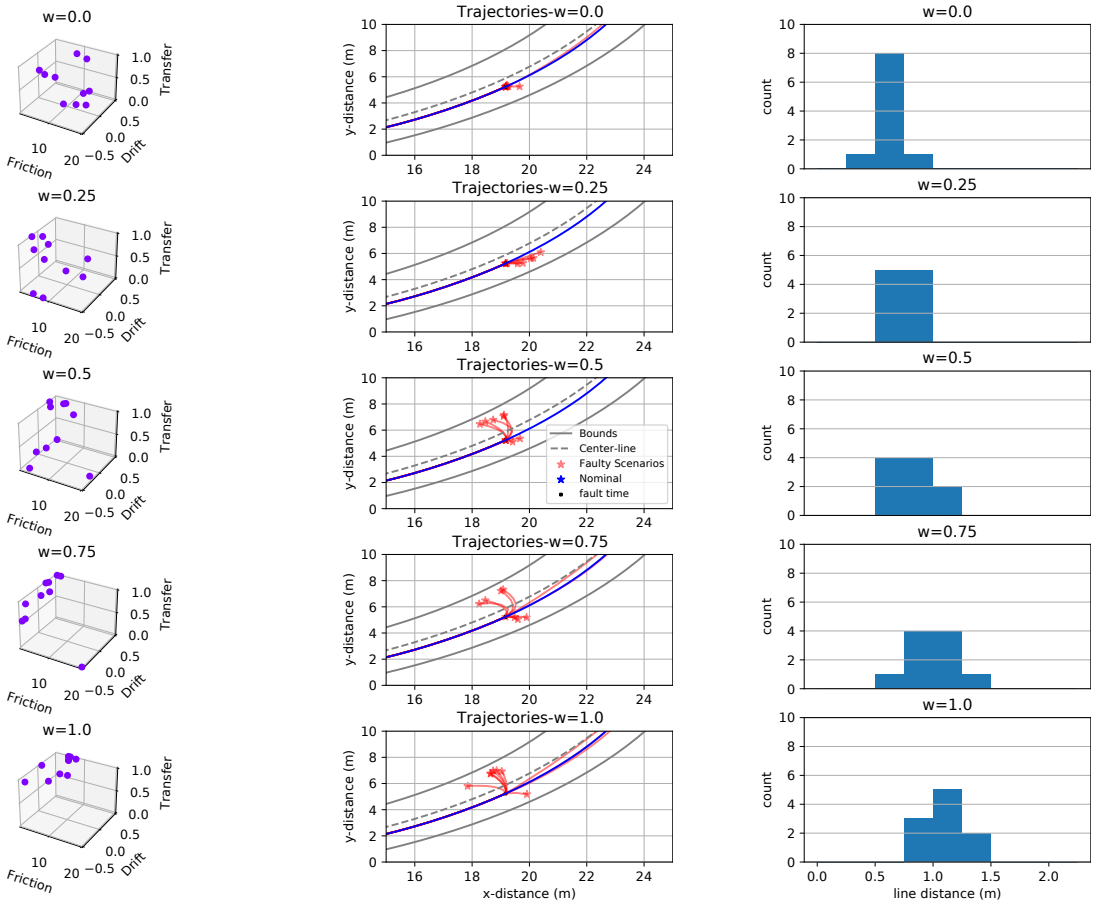
In Formulation 1, the overall severity of the modes is optimized along with the uniqueness of the modes in the faulty state-space used to generate the modes. As a multi-objective problem, the weighting of each objective can be chosen to form a more unique or hazardous set. This section shows the results of varying the weights of this problem to show its effect (both quantitatively and qualitatively) on the modes found. Figure 4 shows the revealed pareto front of optimal solutions found over the five chosen weights (0.0, 0.25, 0.5, 0.75, 1.0). As shown, there is a significant trade-off between faulty state uniqueness and total severity of the set which begins at weight $w = 0.5$ and becomes sever above the weight $w = 0.75$.

To compare the sets of modes more qualitatively, Figure 5 shows the modes found at each algorithm (a) in the faulty state-space, (b) in terms of resulting trajectories and (c) in terms of the resulting line distance distribution. As shown, at low weights, the modes are spread out in the faulty state-space, which results in a number of low-severity modes (most of which are clustered by the fault injection location). However, as the weight is increased ($w >= 0.5$), the modes all converge on a single point in the space with maximum line distance value. This is further reflected in the trajectory end-locations, which cluster at a single point above the course at the weight $w = 1.0$. Qualitatively examining the solutions, the weight of $w = 0.5$ seems to best identify points and trajectories with sufficient distance from each other while having a high severity distribution. However, there may be weights within the interval which achieve a better balance. This shows how properly balancing fault mode uniqueness and severity requires weights to be tuned effectively.

C. Formulation 2: Result-Space Distance

In Formulation 2, the overall severity of the modes is optimized along with the uniqueness of the modes, using output states of the simulation for uniqueness. In this case, these faulty states are the end-locations of the rover when the simulation is completed after the fault. To explore the trade-space between objectives, the weighting parameter w was again varied over a range of values $(0, 0.25, 0.5, 0.75, 1.0)$, producing the pareto front shown in Figure 6. As shown, this pareto front has an inflection point around $w = 0.75$, with most of the change in line distance sum happening at smaller weights.

The solution faulty state-space, resulting trajectories, and line distance distributions are further shown in Figure 7. As shown, at low weights the end-locations of the rover are very spread out, while at high weights (0.75 or higher) the end-locations begin to cluster together (similar to Formulation 1) with high line distance. Interestingly, while faulty state-space uniqueness is not explicitly optimized in this formulation, the modes are nevertheless fairly spread out at low weights, which suggests that unique points in the trajectory space correspond to unique points in the faulty state-space (even if the reverse is not true). This formulation seems to also find better trade-offs between solution distance and



(a) Faulty State-Space. **(b) Resulting Trajectories.** **(c) Resulting Line Distances.**

Fig. 5 Fault scenarios revealed at different weights for Line Distance and State-Space Distance in Formulation 1

severity at $w = 0.5$ and $w = 0.25$, resulting in many different end-locations points that also happen to have a high line distance. This is a better result than what was returned by formulation 1, where the set was composed of a cluster of high-severity end-locations offset by a single far-off point.

V. Discussion

This work explored (1) the formulation of an optimization problem to search the faulty state-space for a tractable set of hazardous modes and (2) the solution of this problem with a cooperative co-evolution algorithm. As shown in Figure 3, this algorithm performs well compared to less-sophisticated optimization approaches on this problem, in part because of its separation between the permutation and simulation of modes, and combination and evaluation of them in the set. However, as an overall strategy for finding modes, it should be noted that previous work performing an exhaustive search of modes (within a certain resolution) took less computational time overall [23], and there is thus a question of the value of this approach. We propose a few advantages: First, the exhaustive search approach can only be carried out at a specific resolution, which may miss worst-case modes which occur in between points. This search approach, on the other hand, progressively searches near promising points (via mutation), resulting in the identification of higher-severity modes. Second, while the computational cost of these experiments is higher than the exhaustive search with given resolution, much of this is because the optimization was run towards completion–if the optimization was cut off at a generation with equal computational cost, the modes in the set would still be more targeted towards the objectives than would be found in the exhaustive search. Finally, it should be noted that in the CCEA approach, the modes are being evaluated intelligently (simulated on creation, rather than at objective evaluation), meaning that the algorithm itself does not add substantial computational cost to the elaboration of modes compared to the fault simulation.

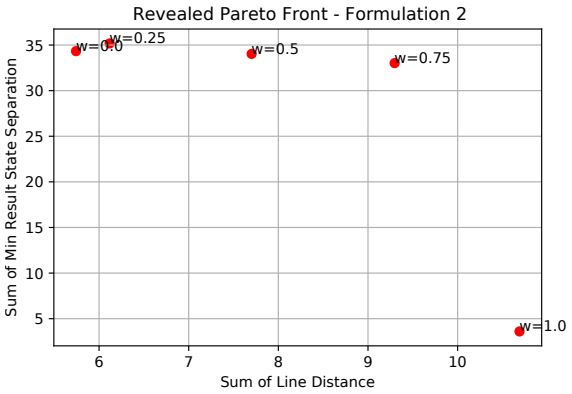


Fig. 6 Found Pareto front of Line Distance and Final Point distance in Formulation 2

Thus, for the task of elaborating a given number of modes in the faulty state-space, we would expect the slight increase in computational cost of running the CCEA algorithm (and saving the population over time) to be offset by its ability to find much higher-impact modes.

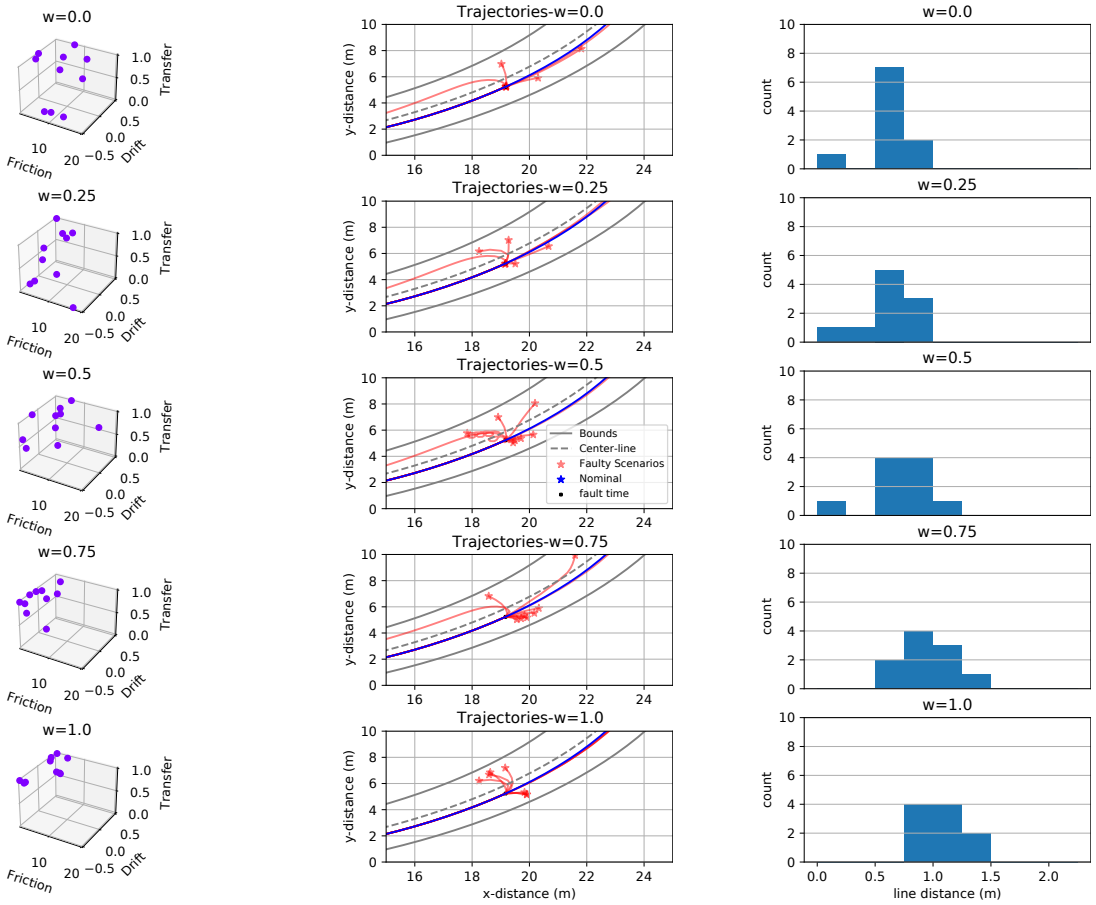
The study of Formulation 1 and 2 in the rover model shows the importance of selecting good weights for the objectives to best balance mode uniqueness and severity. Typically, the weighted sum formulation requires scaling objectives over utopia points so that the objectives are weighted equally at $w = 0.5$. While this was performed in the case study, the results for Formulation 1 (see Figure 5) indicate a difficulty in satisfactorily balancing objectives. Additionally, trying to find a satisfactory balance of objectives by re-running the optimization over each weight introduces a significant computational burden to the process. Future work should thus adapt the CCEA to more naturally accommodate multiple objectives (e.g., by keeping the set of non-dominated solutions over time and using domination for fitness evaluation and mode selection).

Finally, from the design perspective, both formulations (optimizing over the distribution of a faulty state space and result state space) provide important information which can be used to increase system resilience. Optimizing over the faulty state-space (formulation 1) can help ensure there is more exploration of instantiating modes, which may help to promote the discovery of failure causes which would be unknown otherwise (even if these result in similar effects). This information can then be used to better create safegaurds on the system state (e.g., warnings or operational limits) to prevent these faulty modes from occurring in operation. Conversely, optimizing over the result state space (formulation 2) can help designers identify a wide range of potential outcomes that the system may face during its operation. In turn, this information can similarly be used to create general fault-tolerant control policies which actively avoid prevent these outcomes from occurring post-fault (e.g., fault corrections, etc.). Thus, both approaches have use for promoting system resilience, both in terms of identifying unique failure causes (formulation 1) and unique failure effects (formulation 2).

VI. Conclusion

Formulating the generation of a set of scenarios as a multi-objective problem solves one of the major limitations of existing scenario generation approaches–convergence on a single "worst-case" scenario (or type of scenario) rather than revealing the different discrete types of failures that could occur. Within this formulation of the problem, the cooperative co-evolutionary algorithm appears to be suited to the generation of the set because it separates the evaluation and optimization of scenario severity from the formation of the set, resulting in better computational performance than a traditional evolutionary algorithm. However, as demonstrated here, the performance of the overall multi-objective optimization approach is highly sensitive to the underlying formulation of objectives (i.e., the metrics and their weights). While this enables one to tailor the search to the goals of the analysis (i.e., by weighting objectives differently to generate a more unique or severe set), it also creates difficulties for implementation, since one may not know up-front what set of weights may generate the most informative results.

To resolve this limitation, future work should explore the use of non-dominated sorting in the CCEA to enable the search of the entire pareto front at once, rather than a single distinct set of weights. This would make it easier to find the desired balance between set uniqueness and overall severity without needing to re-optimize over different weights. Additionally, while this approach was developed to generate a tractable set of unique high-severity modes, there is an opportunity to use the entire populations of scenarios generated and evaluated during optimization for a larger-scale



(a) Faulty State-Space. **(b) Resulting Trajectories.** **(c) Resulting Line Distances.**

Fig. 7 Fault scenarios revealed at different weights for Line Distance and Final Point distance in Formulation 2

analysis, since these populations span a much larger space than the found solution while still being more targeted (and thus higher severity) than an exhaustive search performed at similar computational cost. Future work should additionally study the use of the populations to explore the faulty state-space and compare this search with more exhaustive search methods previously used in terms of the comparative properties of the modes found. Finally, the approach generated was a static scenario generation approach, only considering the effects of events that occur at a single timestep, rather than chains of events that occur over multiple timesteps (as is considered in stress testing formulations [59]). Future work thus should adapt this approach to the dynamic scenario generation use-case e.g., by using the co-evolutionary approach to train the learning agents used in the algorithms to find a set of unique severe scenarios.

VII. Acknowledgement

This research was partially conducted at NASA Ames Research Center. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government. The United States Government retains, and by accepting the article for publication, the publisher acknowledges that the United States Government retains, a non-exclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for United States Government purposes.

References

[1] Taleb, N. N., *The black swan: The impact of the highly improbable*, Vol. 2, Random house, 2007.

[2] Aven, T., "On the meaning of a black swan in a risk context," *Safety science*, Vol. 57, 2013, pp. 44–51. https://doi.org/https://doi.org/10.1016/j.ssci.2013.01.016.

[3] Craighead, S., "Stress and Resiliency Testing: Mandelbrotian Grey Swan Scenarios," *Enterprise Risk Management Symposium Monograph*, 2011, pp. 1–35.

[4] Aven, T., "Implications of black swans to the foundations and practice of risk assessment and management," *Reliability Engineering & System Safety*, Vol. 134, 2015, pp. 83–91. https://doi.org/https://doi.org/10.1016/j.ress.2014.10.004.

[5] Stamatis, D. H., *Failure mode and effect analysis: FMEA from theory to execution*, Quality Press, 2003.

[6] Ma, X.-Y., Sun, Y.-Z., and Fang, H.-L., "Scenario generation of wind power based on statistical uncertainty and variability," *IEEE Transactions on Sustainable Energy*, Vol. 4, No. 4, 2013, pp. 894–904. https://doi.org/10.1109/TSTE.2013.2256807.

[7] Camal, S., Teng, F., Michiorri, A., Kariniotakis, G., and Badesa, L., "Scenario generation of aggregated Wind, Photovoltaics and small Hydro production for power systems applications," *Applied Energy*, Vol. 242, 2019, pp. 1396–1406. https://doi.org/https://doi.org/10.1016/j.apenergy.2019.03.112.

[8] Ge, L., Liao, W., Wang, S., Bak-Jensen, B., and Pillai, J. R., "Modeling daily load profiles of distribution network for scenario generation using flow-based generative network," *IEEE Access*, Vol. 8, 2020, pp. 77587–77597. https://doi.org/10.1109/ACCESS.2020.2989350.

[9] Soares, J., Borges, N., Ghazvini, M. A. F., Vale, Z., and de Moura Oliveira, P. B., "Scenario generation for electric vehicles' uncertain behavior in a smart city environment," *Energy*, Vol. 111, 2016, pp. 664–675. https://doi.org/https://doi.org/10.1016/j.energy.2016.06.011.

[10] Seljom, P., Kvalbein, L., Hellemo, L., Kaut, M., and Ortiz, M. M., "Stochastic modelling of variable renewables in long-term energy models: Dataset, scenario generation & quality of results," *Energy*, Vol. 236, 2021, p. 121415. https://doi.org/https://doi.org/10.1016/j.energy.2021.121415.

[11] Lei, Y., Wang, D., Jia, H., Chen, J., Li, J., Song, Y., and Li, J., "Multi-objective stochastic expansion planning based on multi-dimensional correlation scenario generation method for regional integrated energy system integrated renewable energy," *Applied energy*, Vol. 276, 2020, p. 115395. https://doi.org/https://doi.org/10.1016/j.apenergy.2020.115395.

[12] Austgen, B., Garcia, M., Pierre, B., Hasenbein, J., and Kutanoglu, E., "Winter storm scenario generation for power grids based on historical generator outages," *2022 IEEE/PES Transmission and Distribution Conference and Exposition (T&D)*, IEEE, 2022, pp. 1–5. https://doi.org/10.1109/TD43745.2022.9816863.

[13] Trakas, D. N., and Hatziargyriou, N. D., "Optimal distribution system operation for enhancing resilience against wildfires," *IEEE Transactions on Power Systems*, Vol. 33, No. 2, 2017, pp. 2260–2271. https://doi.org/10.1109/TPWRS.2017.2733224.

[14] Trakas, D. N., Panteli, M., Hatziargyriou, N. D., and Mancarella, P., "Spatial risk analysis of power systems resilience during extreme events," *Risk Analysis*, Vol. 39, No. 1, 2019, pp. 195–211. https://doi.org/10.1111/risa.13220.

[15] Herman, J. D., Zeff, H. B., Lamontagne, J. R., Reed, P. M., and Characklis, G. W., "Synthetic drought scenario generation to support bottom-up water supply vulnerability assessments," *Journal of Water Resources Planning and Management*, Vol. 142, No. 11, 2016, p. 04016050. https://doi.org/https://doi.org/10.1061/(ASCE)WR.1943-5452.0000701.

[16] Faturechi, R., Levenberg, E., and Miller-Hooks, E., "Evaluating and optimizing resilience of airport pavement networks," *Computers & Operations Research*, Vol. 43, 2014, pp. 335–348. https://doi.org/https://doi.org/10.1016/j.cor.2013.10.009.

[17] Short, A.-R., and DuPont, B. L., "Computational Cognition for Mission Command and Control Decisions Facing Risk in Unknown Environments," *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Vol. 59193, American Society of Mechanical Engineers, 2019, p. V02BT03A020. https://doi.org/https://doi.org/10.1115/DETC2019-98483.

[18] Balchanos, M., Li, Y., and Mavris, D., "Towards a method for assessing resilience of complex dynamical systems," *2012 5th International Symposium on Resilient Control Systems*, IEEE, 2012, pp. 155–160. https://doi.org/10.1109/ISRCS.2012.6309310.

[19] McIntire, M. G., Keshavarzi, E., Tumer, I. Y., and Hoyle, C., "Functional models with inherent behavior: Towards a framework for safety analysis early in the design of complex systems," *ASME International Mechanical Engineering Congress and Exposition*, Vol. 50657, American Society of Mechanical Engineers, 2016, p. V011T15A035. https://doi.org/https://doi.org/10.1115/IMECE2016-67040.

[20] Irshad, L., Demirel, H. O., and Tumer, I. Y., "Automated generation of fault scenarios to assess potential human errors and functional failures in early design stages," *Journal of computing and information science in engineering*, Vol. 20, No. 5, 2020. https://doi.org/https://doi.org/10.1115/1.4047557.

[21] Nejad, H., and Mosleh, A., "Automated risk scenario generation using system functional and structural knowledge," *ASME International Mechanical Engineering Congress and Exposition*, Vol. 42304, 2005, pp. 85–89. https://doi.org/https://doi.org/10.1115/IMECE2005-81331.

[22] Chen, Y., Yang, S., and Men, W., "Automatic Generation of Failure Mechanism Propagation Scenario via Guided Simulation and Intelligent Algorithm," *IEEE Access*, Vol. 7, 2019, pp. 34762–34775. https://doi.org/10.1109/ACCESS.2019.2904305.

[23] Hulse, D., and Irshad, L., "Synthetic fault mode generation for resilience analysis and failure mechanism discovery," *Journal of Mechanical Design*, 2022, pp. 1–10. https://doi.org/10.1115/1.4056320, URL https://doi.org/10.1115/1.4056320.

[24] Corso, A., Du, P., Driggs-Campbell, K., and Kochenderfer, M. J., "Adaptive stress testing with reward augmentation for autonomous vehicle validation," *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2019, pp. 163–168. https://doi.org/10.1109/ITSC.2019.8917242.

[25] Oliveira, B. B., Carravilla, M. A., and Oliveira, J. F., "A diversity-based genetic algorithm for scenario generation," *European Journal of Operational Research*, Vol. 299, No. 3, 2022, pp. 1128–1141. https://doi.org/https://doi.org/10.1016/j.ejor.2021.09.047.

[26] Römisch, W., *Stochastic Programming, Scenario Generation In*, John Wiley & Sons, Ltd, 2018, pp. 1–5. https://doi.org/https://doi.org/10.1002/9781118445112.stat08075, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118445112.stat08075.

[27] Kaut, M., and Stein, W., *Evaluation of scenario-generation methods for stochastic programming*, Humboldt-Universität zu Berlin, Mathematisch-Naturwissenschaftliche Fakultät . . . , 2003. https://doi.org/https://doi.org/10.18452/8296.

[28] Löhndorf, N., "An empirical analysis of scenario generation methods for stochastic optimization," *European Journal of Operational Research*, Vol. 255, No. 1, 2016, pp. 121–132. https://doi.org/https://doi.org/10.1016/j.ejor.2016.05.021.

[29] Messina, E., and Toscani, D., "Hidden Markov models for scenario generation," *IMA Journal of Management Mathematics*, Vol. 19, No. 4, 2008, pp. 379–401. https://doi.org/10.1093/imaman/dpm026.

[30] Høyland, K., Kaut, M., and Wallace, S. W., "A heuristic for moment-matching scenario generation," *Computational optimization and applications*, Vol. 24, No. 2, 2003, pp. 169–185. https://doi.org/https://doi.org/10.1023/A:1021853807313.

[31] Casey, M. S., and Sen, S., "The scenario generation algorithm for multistage stochastic linear programming," *Mathematics of Operations Research*, Vol. 30, No. 3, 2005, pp. 615–631. https://doi.org/https://doi.org/10.1287/moor.1050.0146.

[32] Guastaroba, G., Mansini, R., and Speranza, M. G., "On the effectiveness of scenario generation techniques in single-period portfolio optimization," *European Journal of Operational Research*, Vol. 192, No. 2, 2009, pp. 500–511. https://doi.org/https://doi.org/10.1016/j.ejor.2007.09.042.

[33] Abdelgawad, A. A., Noori, N., and Comes, T., "Automated scenario generation for training of humanitarian responders in high-risk settings," *International Journal of Simulation–Systems, Science & Technology*, Vol. 19, No. 5, 2018. URL https://ijssst.info/Vol-19/No-5/paper23.pdf.

[34] Ferdinandus, G. R., "Automated scenario generation, coupling planning techniques with smart objects," Master's thesis, Utrecht University, 2012. URL https://studenttheses.uu.nl/handle/20.500.12932/11826.

[35] Zook, A., Lee-Urban, S., Riedl, M. O., Holden, H. K., Sottilare, R. A., and Brawner, K. W., "Automated scenario generation: toward tailored and optimized military training in virtual environments," *Proceedings of the international conference on the foundations of digital games*, 2012, pp. 164–171. https://doi.org/https://doi.org/10.1145/2282338.2282371.

[36] Rowe, J., Smith, A., Pokorny, B., Mott, B., and Lester, J., "Toward automated scenario generation with deep reinforcement learning in GIFT," *Proceedings of the Sixth Annual GIFT User Symposium*, 2018, pp. 65–74.

[37] Chatterji, G. B., Palopo, K., Zheng, Y., and Nguyen, J., "Automated Scenario Generation for Human-in-the-Loop Simulations," *2018 Modeling and Simulation Technologies Conference*, 2018, p. 3751. https://doi.org/https://doi.org/10.2514/6.2018-3751.

[38] Chatterji, G. B., and Zheng, Y., "Automated Scenario Generation for Meeting Human-in-the-Loop Simulation Requirements," *AIAA Scitech 2019 Forum*, 2019, p. 1710. https://doi.org/https://doi.org/10.2514/6.2019-1710.

[39] Anggreeni, I., Van der Voort, M., et al., "Classifying Scenarios in a Product Design Process: a study towards semi-automated scenario generation," *CIRP design conference 2008*, 2008.

[40] Pedersen, T. A., Neverlien, Å., Glomsrud, J. A., Ibrahim, I., Mo, S. M., Rindarøy, M., Torben, T., and Rokseth, B., "Evolution of Safety in Marine Systems: From System-Theoretic Process Analysis to Automated Test Scenario Generation," *Journal of Physics: Conference Series*, Vol. 2311, IOP Publishing, 2022, p. 012016. https://doi.org/10.1088/1742-6596/2311/1/012016.

[41] Pan, X., Hu, L., Xin, Z., Zhou, S., Lin, Y., and Wu, Y., "Risk scenario generation based on importance measure analysis," *Sustainability*, Vol. 10, No. 9, 2018, p. 3207. https://doi.org/https://doi.org/10.3390/su10093207.

[42] Chowdhury, N. S., and Choi, Y., "Automated Scenario Generation for Model Checking Trampoline Operating System," *Proceedings of the Korea Information Processing Society Conference*, Korea Information Processing Society, 2011, pp. 1342–1345.

[43] Sapna, P., and Mohanty, H., "Automated scenario generation based on uml activity diagrams," *2008 International Conference on Information Technology*, IEEE, 2008, pp. 209–214. https://doi.org/10.1109/ICIT.2008.52.

[44] Maiden, N. A., Minocha, S., Manning, K., and Ryan, M., "CREWS-SAVRE: Systematic scenario generation and use," *Proceedings of IEEE International Symposium on Requirements Engineering: RE'98*, IEEE, 1998, pp. 148–155. https://doi.org/10.1109/ICRE.1998.667820.

[45] Nalic, D., Mihalj, T., Bäumler, M., Lehmann, M., Eichberger, A., and Bernsteiner, S., "Scenario based testing of automated driving systems: A literature survey," *FISITA web Congress*, 2020, pp. 1–10. https://doi.org/10.46720/f2020-acm-096.

[46] Ding, W., Xu, C., Arief, M., Lin, H., Li, B., and Zhao, D., "A Survey on Safety-Critical Driving Scenario Generation-A Methodological Perspective." *CoRR*, 2022. https://doi.org/https://doi.org/10.48550/arXiv.2202.02215.

[47] Koren, M., Corso, A., and Kochenderfer, M. J., "The adaptive stress testing formulation," *arXiv preprint arXiv:2004.04293*, 2020. https://doi.org/https://doi.org/10.48550/arXiv.2004.04293.

[48] Xie, Y., Dai, K., and Zhang, Y., "A Real-time Critical-scenario-generation Framework for Testing Autonomous Driving System," *arXiv preprint arXiv:2206.00910*, 2022. https://doi.org/https://doi.org/10.48550/arXiv.2206.00910.

[49] Moss, R. J., Lee, R., Visser, N., Hochwarth, J., Lopez, J. G., and Kochenderfer, M. J., "Adaptive stress testing of trajectory predictions in flight management systems," *2020 AIAA/IEEE 39th Digital Avionics Systems Conference (DASC)*, IEEE, 2020, pp. 1–10. https://doi.org/10.1109/DASC50938.2020.9256730.

[50] Lee, R., Kochenderfer, M. J., Mengshoel, O. J., Brat, G. P., and Owen, M. P., "Adaptive stress testing of airborne collision avoidance systems," *2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC)*, IEEE, 2015, pp. 6C2–1. https://doi.org/10.1109/DASC.2015.7311450.

[51] Potter, M., and De Jong, K., "Cooperative coevolution: an architecture for evolving coadapted subcomponents," *Evolutionary computation*, 2000, p. 1–29. https://doi.org/https://doi.org/10.1162/106365600568086.

[52] Boucher, D. H., James, S., and Keeler, K. H., "The Ecology of Mutualism," *Annual Review of Ecology and Systematics*, Vol. 13, 1982, pp. 315–347. URL http://www.jstor.org/stable/2097071.

[53] Ma, X., Li, X., Zhang, Q., Tang, K., Liang, Z., Xie, W., and Zhu, Z., "A Survey on Cooperative Co-Evolutionary Algorithms," *IEEE Transactions on Evolutionary Computation*, Vol. 23, No. 3, 2019, pp. 421–441. https://doi.org/10.1109/TEVC.2018.2868770.

[54] Soria Zurita, N. F., Colby, M. K., Tumer, I. Y., Hoyle, C., and Tumer, K., "Design of complex engineered systems using multi-agent coordination," *Journal of Computing and Information Science in Engineering*, Vol. 18, No. 1, 2018. https://doi.org/https://doi.org/10.1115/1.4038158.

[55] Panait, L., "The Analysis and Design of Concurrent Learning Algorithms for Cooperative Multiagent Systems," Ph.D. thesis, George Mason University, 2006.

[56] Iscen, A., Agogino, A., SunSpiral, V., and Tumer, K., "Controlling tensegrity robots through evolution," *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, 2013, pp. 1293–1300. https://doi.org/https://doi.org/10.1145/2463372.2463525.

[57] Colby, M., and Tumer, K., "Fitness function shaping in Multiagent Cooperative Coevolutionary algorithms," *Autonomous Agents and Multi-Agent Systems*, Vol. 31, No. 2, 2015, p. 179–206. https://doi.org/10.1007/s10458-015-9318-0.

[58] Ma, X., Li, X., Zhang, Q., Tang, K., Liang, Z., Xie, W., and Zhu, Z., "A Survey on Cooperative Co-Evolutionary Algorithms," *IEEE Transactions on Evolutionary Computation*, Vol. 23, No. 3, 2019, pp. 421–441. https://doi.org/10.1109/TEVC.2018.2868770.

[59] Koren, M., Corso, A., and Kochenderfer, M. J., "The Adaptive Stress Testing Formulation," , 2020. https://doi.org/10.48550/ARXIV.2004.04293, URL https://arxiv.org/abs/2004.04293.