Abstract—This work is in the field of requirements driven search-based test case generation methods for Cyber-Physical Systems (CPS). The basic characteristic of search-based testing methods is that the search process is guided by high level requirements captured in formal logic and, in particular, Signal Temporal Logic (STL). Given a system trajectory, STL specifications can be equipped with quantitative semantics which evaluate the closeness of the given trajectory from violating the requirement. Hence, by searching for trajectories of decreasing value with respect to the specification, a test generation method can be formulated which searches for system behaviors with a closeness to violation value of less than 0. These system behaviors, i.e., trajectories that violate the requirements and yield STL closeness value less than 0, are referred to as falsifying behaviors. In addition, signed distance can be utilized when searching for trajectories that maximally violate the specification (negative specification valuations). In this work, we propose the use of a stochastic search method that mixes global and local search for system test case generation. The implemented search method models input-output relationships between test cases and the observed STL closeness values of the yielded system trajectories, adaptively linking input-out of both global and local regional modeling. The method shows improved finite time performance, i.e., quick identification of falsification behaviors, over current search-based test case generation methods. Further, given no falsifying behaviors are found in finite time our method is capable of quantifying the certainty that no falsifying behaviors exist.

I. INTRODUCTION

Falsification refers to any method that attempts to demonstrate that a requirement is falsified (it is not true) on a system. The falsification problem is prominent in the domain of Cyber-Physical Systems (CPS) since the majority of such systems are safety critical. Hence, any behavior which violates system requirements must be detected early on and be corrected. In the simplest possible case, the requirement is to avoid an unsafe set and, thus, the falsification problem reduces to searching for a path from a set of initial conditions to the unsafe set [1], [2]. If no path exists to an unsafe set, then the conclusion is that the CPS is safe.

More complex requirements can be specified in Signal Temporal Logic (STL) [3] which can specify sequences of unsafe states with timing requirements or reactive requirements. Motivated by the need to falsify more complex requirements in STL, Nghiem et. al. [4] presented a framework where the falsification problem is translated into a minimization problem through the notion of robustness for STL [5], [6]. Temporal logic robustness is a quantitative measure which captures how robustly a system behavior satisfies a requirement, i.e. the robustness of a given system trajectory is a measure of how close the system came to reaching to falsifying the specified requirement. Large positive values mean that the behavior is robustly correct (the behavior is very far from violating the STL), while large negative values imply that the behavior is robustly unsafe (the behavior is very far from not violating the STL). The resulting minimization problem searches for points (or signals) in the input and parameter space of the system such that the resulting temporal logic robustness is less than or equal to zero (negative), thus falsifying the STL.

Since the problem reduces to an optimization problem, substantial research effort has been devoted to repurposing existing optimization methods or, even, developing new ones which take into account the structure of the problem. In this work we tailor an existing optimization framework to the problem of falsification. The optimization framework adaptively iterates between global and local stochastic search techniques. The global search producing a prediction to guide initialization of a local search- which discovers improving regions by dynamically focusing sampling in small regions. Further, if no falsifications are found our approach provides confidence estimates of the safety of the system.

Contributions: We implement an instance of the Stochastic Optimization with Adaptive Restart (SOAR) framework [7] in the S-TaLiRo tool [8]. The application of SOAR for falsification not only shows better falsification rates than existing benchmarking techniques, but also provides confidence intervals over the predicted minimum system robustness when the system is unable to be observably falsified. This is the first time a falsification method provides uncertainty quantification on the minimum predicted robustness. This is an important benefit for practitioners since they can now decide whether further testing is needed.

II. RELEVANT LITERATURE

A. Falsification in Cyber-Physical Systems

Due to the importance of the falsification problem to safety critical CPS, there has been a spur of research in the area of falsification leading to many different search methods [9] as well as specification formalisms [3]. Since the STL falsification problem can be reduced to an optimization problem [10], many new optimization methods have been proposed which focus on the falsification problem, e.g.,
Cross-Entropy Optimization [11]. More recently, there have been several new approaches to the problem such as Monte Carlo Tree Search [12] or Deep Reinforcement Learning [13]. Closer to our own work, the authors in [14] present a combination of local and global search which, in addition, uses support vector machines to identify promising regions of the search space for further sampling. Our approach with SOAR is similar with a local and global search, but, instead of classifying regions as promising we dynamically focus local sampling into appealing regions and additionally provide a prediction of robustness throughout the input space.

B. Stochastic Search Methods

It is important to highlight that optimization algorithms are stochastic when: (1) an observed objective function evaluation is subject to noise, or (2) randomness is injected via the search procedure itself while the function results are noiseless [15], or a mixture of the two. We assume noiseless (deterministic) evaluations, as is common in cyber-physical system representations, and thus review literature focusing on (2), the aspect of injected randomness, rather than the way noisy evaluations are handled.

Among recent local optimization approaches, Stochastic Trust-Region Response-Surface Method (STRONG) iteratively executes two stages: (1) constructing local models through linear regression, when gradient information is not readily available, and (2) formulating and solving the associated trust-region subproblem [16]. In [17], the proposed Derivative Free Adaptive Sampling Trust Region Optimization (ASTRO-DF) uses derivative free trust-region methods to generate and statistically certify local stochastic models, that are used to update candidate solution values through an adaptive sampling scheme.

Without assumptions on the degree of non-linearity in the objective function, the drawback of these methods is that the quality of the discovered local minimum may be poor, relative to the entire surface. Research has been conducted on multi-starts and restart policies which have overcome these issues in some cases [18].

Global algorithms aim to balance the trade-off between exploration and exploitation, and investigate un-sampled regions without the promise of improvement. Notable exceptions exist. One intelligent extension of such a method is Simulated Annealing, which makes use of underlying Markov Chains to adaptively bias search towards improving locations. Though, in practice, it is necessary to sample from the falsifying level set within a finite amount of time/testing budget. One naïve sampling scheme that yields asymptotic guarantees over compact sets \( \mathcal{X} \) is uniform random sampling, and one intelligent extension of such a method is Simulated Annealing, which makes use of underlying Markov Chains to adaptively bias search towards improving locations. Though, given no falsifications were sampled, neither of these have the capability to provide confidence that no falsifying solutions exist.

In this work we apply the SOAR framework, which has been shown to effectively identify global optima of highly non-linear surfaces in moderately-high problem dimensions [7]. The SOAR framework is comprised of three interconnected components: (1) a global meta-model used to sample across \( \mathcal{X} \), (2) a local search method to exploit and model regions of \( \mathcal{X} \), and (3) a mechanism to determine which mode to sample in that dynamically reacts to local search progression and adaptively updates and samples from

\[
\begin{align*}
\mathbf{x}^* & \in \underset{\mathbf{x} \in \mathcal{X}}{\arg\min} f(\mathbf{x}) \\
& \text{(1)}
\end{align*}
\]

where \( \mathcal{X} \subseteq \mathbb{R}^d \) represents the searchable region of inputs, including initial conditions and input signals to the system that parameterized through \( d_i \) initial conditions and \( d_f \) signal control points, and \( f(\mathbf{x}) \) is the robustness of the trajectory yielded by the input \( \mathbf{x} \). Through this perspective a system can be verified if the globally minimal robustness is positive, i.e., \( f(\mathbf{x}^*) > 0 \), and if \( f(\mathbf{x}) < 0 \) the trajectory \( \mathbf{x} \) is falsifying. Unique to the falsification problem, the optimization procedure can be stopped prematurely if a sampled location \( \mathbf{x}_f \) is falsifying. The primary goal of is to efficiently find a falsifying sample:

\[
\begin{align*}
\mathbf{x} \in \mathcal{X} : f(\mathbf{x}) & \leq 0. \\
& \text{(2)}
\end{align*}
\]

or to provide confidence that no such falsifying solution exist in \( \mathcal{X} \). In theory we desire sampling algorithms to converge to a globally minimal robustness value \( f(\mathbf{x}^*) \) thereby asymptotically validating system falsification/verification. However, in practice, it is necessary to sample from the falsifying level set within a finite amount of time/testing budget. One naïve sampling scheme that yields asymptotic guarantees over compact sets \( \mathcal{X} \) is uniform random sampling, and one intelligent extension of such a method is Simulated Annealing, which makes use of underlying Markov Chains to adaptively bias search towards improving locations. Though, given no falsifications were sampled, neither of these have the capability to provide confidence that no falsifying solutions exist.

Several proposed approaches aim to bridge the gap between local and global methods. Industrial Strength Compass (ISC) [22] finds solution seeds via a global Niched Genetic Algorithm to facilitate a quick multi-start of the Convergent Optimization via Most-Promising-Area Stochastic Search (COMPASS)- an effective partition based search. The Balanced Explorative and Exploitative Search with Estimation (BESEE) framework [23] balances (global) exploratory and (local) exploitative behavior at each iteration by sampling with probability \( p \) from a global distribution, and with probability \( 1 - p \) from distribution chosen from a user-defined local distribution family. The parameter \( p \) is chosen adaptively every \( k \) function evaluations, weighing a calculated improvement score against an a priori threshold value. The approach this work employs is the SOAR framework [7]; SOAR adaptively selects between global and local search via observed search progression, and iteratively transfers learned information between the two search modes.

III. APPLICATION OF SOAR FOR FALSIFICATION

We approach the falsification problem by aiming to solve the more general global optimization problem:

\[
\begin{align*}
\mathbf{x}^* & \in \underset{\mathbf{x} \in \mathcal{X}}{\arg\min} f(\mathbf{x}) \\
& \text{(1)}
\end{align*}
\]

where \( \mathcal{X} \subseteq \mathbb{R}^d \) represents the searchable region of inputs, including initial conditions and input signals to the system that parameterized through \( d_i \) initial conditions and \( d_f \) signal control points, and \( f(\mathbf{x}) \) is the robustness of the trajectory yielded by the input \( \mathbf{x} \). Through this perspective a system can be verified if the globally minimal robustness is positive, i.e., \( f(\mathbf{x}^*) > 0 \), and if \( f(\mathbf{x}) < 0 \) the trajectory \( \mathbf{x} \) is falsifying. Unique to the falsification problem, the optimization procedure can be stopped prematurely if a sampled location \( \mathbf{x}_f \) is falsifying. The primary goal of is to efficiently find a falsifying sample:

\[
\begin{align*}
\mathbf{x} \in \mathcal{X} : f(\mathbf{x}) & \leq 0. \\
& \text{(2)}
\end{align*}
\]

or to provide confidence that no such falsifying solution exist in \( \mathcal{X} \). In theory we desire sampling algorithms to converge to a globally minimal robustness value \( f(\mathbf{x}^*) \) thereby asymptotically validating system falsification/verification. However, in practice, it is necessary to sample from the falsifying level set within a finite amount of time/testing budget. One naïve sampling scheme that yields asymptotic guarantees over compact sets \( \mathcal{X} \) is uniform random sampling, and one intelligent extension of such a method is Simulated Annealing, which makes use of underlying Markov Chains to adaptively bias search towards improving locations. Though, given no falsifications were sampled, neither of these have the capability to provide confidence that no falsifying solutions exist.

In this work we apply the SOAR framework, which has been shown to effectively identify global optima of highly non-linear surfaces in moderately-high problem dimensions [7]. The SOAR framework is comprised of three interconnected components: (1) a global meta-model used to sample across \( \mathcal{X} \), (2) a local search method to exploit and model regions of \( \mathcal{X} \), and (3) a mechanism to determine which mode to sample in that dynamically reacts to local search progression and adaptively updates and samples from

\[
\begin{align*}
\mathbf{x}^* & \in \underset{\mathbf{x} \in \mathcal{X}}{\arg\min} f(\mathbf{x}) \\
& \text{(1)}
\end{align*}
\]
the global model. (1) The global meta-model expresses information learned from observed input-robustness relations to estimate a robustness, \( \hat{f}(x) \), for all \( x \in \mathcal{X} \); enabling balanced sampling between inputs with uncertain \( f(x) \) and inputs with minimal \( f(x) \). In this implementation we make use of a Gaussian process (GP) model with estimation equations presented in the Section III-A. (2) The local search serves to thoroughly exploit and precisely model small regions of inputs, discovering local features of the robustness surface. Our implemented local search uses a trust region methodology to focus modeling and sampling into localized regions, the details are also given in Section III-A. (3) The interconnection of the two sampling schemes facilitates meaningful sharing of robustness information between local and global approaches, while dynamically determining which mode of sampling should be used. The details on the switching mechanism between local and global is given in Section III-B.

### A. Global and Local Modeling and Sampling

**Meta-Model Estimation:** Observed input-output robustness relationships are captured through a GP model fit to the pairs \((x, f(x))\), assuming that the underlying response surface can be expressed as a realization of \( F(x) = \mu(x) + Z(x) \). For simplicity, let \( \mu(x) = \mu \), assessing a constant mean throughout \( \mathcal{X} \). Resultantly \( Z(x) \) is a zero mean GP defined by the process variance \( \tau \) and spatial correlation matrix \( R_x \), \( Z(x) \sim GP(\mu, \tau^2R_x) \). Fitting \( n \) input-robustness pairs, each \((i, j)\) element of the \( n \times n \) matrix \( R_x \) is calculated with Gaussian correlation function: \( R_{i:j} = \prod_{l=1}^{d} \exp(-\phi(x_{i,l} - x_{j,l}))^2 \). Where \( \phi \) is the \( d \)-dimensional vector of dimensional smoothing hyper parameters, and \( x_{i,t} \) is the \( t \)-th dimension of observation \( x_i \). Given the \( n \times 1 \) vector \( f \) of robustness observations, the maximum likelihood estimators of the two GP parameters are [24]:

\[
\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} f_i \quad \text{and} \quad \tau^2 = \frac{1}{n} \sum_{i=1}^{n} (f - \hat{\mu})^T (f - \hat{\mu})
\]

Resulting in a best unbiased predictor for any \( x_0 \in \mathcal{X} \) of:

\[
\hat{f}(x_0) = \hat{\mu} + r^T \tau^{-1} (f - \hat{\mu})
\]

with a corresponding predictive variance of:

\[
Var[\hat{f}(x_0)] = s^2(x_0) = \tau^2 \left( 1 - r^T \tau^{-1} r + \frac{(1 - r^T \tau^{-1} r)^2}{\tau^2} \right)
\]

where \( r \) is the \( n \times 1 \) vector of Gaussian correlations between \( x_0 \) and each of the \( n \) previously sampled \( x \) locations, i.e., \( r_i(x_0) = Corr(Z(x_0), Z(x_i)) \).

**Local Trust Region Based Modeling:** To isolate and exploit regions of interest with a local search we adopt a trust region method [25] that restricts the solution space that is modeled and sampled from, allowing regional characteristics to be expressed and exploited. Due to the often non-differentiable nature of the robustness function standard gradient descent local search approaches fail in practice. This is a clear advantage of the trust region method, where local meta-models can be estimated over a region, capturing general regional trends and systematically moving the trust region centroid towards regions with lower robustness. Moreover, the trust region dynamically adjusts in size such that the estimated meta-model achieves accurate robustness improvement predictions. In Section III-B, we show how this adaptive trust region sizing can be used to estimate local search progression rate.

In the SOAR framework there is an inner loop, the iters/subproblems solved in a single local search, and an outer global loop with complete local searches as itterates. As such \( k_\ell \) defines the \( \ell \)-th subproblem (inner loop iterate) of the \( k \)-th local search (out loop iterate) performed in the SOAR framework. At each \( k_\ell \) subproblem a trust region \( \Delta_{s_{k_\ell}}(x_{k_\ell}) \) is defined by the centroid \( x_{k_\ell} \) and a hypercube in \( d \) dimensions with length \( s_{k_\ell} \) between the centroid \( x_{k_\ell} \) and each facet of \( \Delta_{s_{k_\ell}}(x_{k_\ell}) \). A random Latin hypercube space filling design of \( 5 \times d \) points in \( \Delta_{s_{k_\ell}}(x_{k_\ell}) \) is sampled and used to estimate a local GP with predictions \( \hat{f}_{k_\ell}(x) \) for all \( x \in \Delta_{s_{k_\ell}}(x_{k_\ell}) \). This estimated GP is used to propose a new candidate centroid \( x_c \). The sampled value \( f(x_c) \) then determines how the trust region is controlled, i.e., whether the candidate \( x_c \) will become the next centroid \( x_{k_\ell+1} \) and how the trust region size changes. We use a ratio-comparison test statistic \( \rho \) is constructed [25], [16]

\[
\rho = \frac{f(x_{k_\ell}) - f(x_c)}{\hat{f}_{k_\ell}(x_{k_\ell}) - \hat{f}_{k_\ell}(x_c)}.
\]

Comparing \( \rho \) with the user defined threshold values \( \eta_1, \eta_2 \) and \( 0 < \eta_1 \leq \eta_2 < 1 \) (in this work we set \( \eta_1 = 0.25 \) and \( \eta_2 = 0.75 \)) three cases for trust region control arise:

**Case 1** \( \rho \leq \eta_1 \implies \) no translation \( x_{k_\ell+1} \leftarrow x_{k_\ell} \); trust region size reduced \( s_{k_\ell+1} \leftarrow s_{k_\ell} \).

**Case 2** \( \eta_1 < \rho \leq \eta_2 \implies \) centroid translates \( x_{k_\ell+1} \leftarrow x_c \); no size scaling \( s_{k_\ell+1} \leftarrow s_{k_\ell} \).

**Case 3** \( \rho > \eta_2 \implies \) centroid translates \( x_{k_\ell+1} \leftarrow x_c \); grow trust region size \( s_{k_\ell+1} \leftarrow s_{k_\ell} \cdot \gamma \).

Here \( \omega = 0.8 \) and \( \gamma = 1.2 \) are user defined and represent the trust region reduction and growth rates. Thus a small test statistic \( \rho \) reduces the size of the trust region to better modeling capability in the current region, and when it is large we increase the trust region size. After the appropriate control is executed we test whether to remain in or exit the current \( k \)-th local (trust region) search. This testing is explained in Section III-B. If we remain in the local search then we increment \( \ell \leftarrow \ell + 1 \) and return to sample/re-estimate the local GP and solve the newly updated \( k_\ell \leftarrow k_\ell+1 \) subproblem.

**Gaussian Process Based Sampling:** Both global and local sampling is driven by their respective estimated GP. GP’s lend themselves nicely to sampling as they have closed forms for both predictions \( f(x) \), see eq. (3), and prediction variance \( s^2(x) \), see eq. (4). In this work we assign a sampling score to each \( x \in \mathcal{X} \) or \( x \in \Delta_{s_{k_\ell}}(x_{k_\ell}) \) based upon \( \hat{f}(x) \) and \( s^2(x) \). We utilize the common expected improvement sampling criteria [21], [26] which at an arbitrary \( k \)-th iteration,
can be defined as:

\[ EI_k(x) = E \left[ \max \left( \left( f^*_k - f(x) \right) \Phi \left( \frac{f^*_k - f(x)}{s(x)} \right), 0 \right) \right] \]

where \( f^*_k \) is best observed (minimum) robustness observed up to the \( k \)-th iteration. In our local trust region searches we generate candidate centroids via GP based sampling and the above stated expected improvement criteria (6), such that:

\[ x_c \in \arg \max_{x \in \Delta_{k\ell}} EI_{k\ell}(x). \] (7)

For global sampling we use the proposed extension to the expected improvement criteria from [7], namely, the crowded expected improvement \( EI_c(x) \). This extension is shown to be critical as the expected improvement tends to favor exploitation over exploration [27]. While this bias towards exploitation stands to benefit the local trust region searches, it is intuitive that global sampling should prefer exploration as exploitation is done by the local search. The crowded expected improvement samples locations that are promising, with respect to the standard expected improvement criteria, while simultaneously maximizing the distance from all previously observed locations. The crowded expected improvement is formalized as:

\[
EI^c_k(x_i) = \begin{cases} 
\sum_{\ell=1}^{d} \min_{x_j \in S_k} |x_{i,\ell} - x_{j,\ell}|, & EI_k(x_i) \geq \alpha EI^c_k \\
0, & EI_k(x_i) < \alpha EI^c_k 
\end{cases}
\] (8)

where \( S_k \) is the set of all sampled locations up to the \( k \)-th iteration. \( EI^c_k = \max EI_k \) is the maximum expected improvement over all of \( X \), and \( \alpha \in (0, 1) \) defines the \( \alpha \)-level set of locations with best expected improvement. In this work we set \( \alpha = 0.05 \). At iteration \( k \), the global GP based sampling decision is then determined as:

\[ x_k \in \arg \max_{x \in X} EI^c_k(x) \] (9)

B. Global and Local Interconnection and Transition

In the SOAR framework, global and local sampling are connected through an adaptive local search restart procedure and sample sharing. Local sampling is monitored and an indicator for rate of search progress is assessed after each local search iteration. When a local search (the trust region search in this case) is dynamically terminated sampling returns to the global search mode to propose candidate local search restart locations (i.e., initializing trust region centroids), that are either accepted or rejected.

Motivated by practical real-world falsification scenarios, our SOAR implementation considers the limited simulation budget. Thus the primary purpose of local sampling is to focus the search and discover regions of local minima, rather than using large portions of the budget to precisely discover a single local minimum. This leads to an indicator of search progression which balances the necessary budget spent learning a good trust region size with the cost of excessive centroid stalling and marginal robustness improvement. The search progression indicator used in this work (10) is the maximum if the ratio in each dimension of the candidate step vector to the allowed step length, \(|x_{k\ell,i} - x_{e,i}| / s_{k\ell}\), where \( i \in \{1, \ldots, d\} \) represents the dimension of \( x_{k\ell} \) and \( x_e \).

\[ \psi_{k\ell} = \max \left\{ \frac{|x_{k\ell,i} - x_{e,i}|}{s_{k\ell}}, \ldots, \frac{|x_{k\ell,d} - x_{e,d}|}{s_{k\ell}} \right\} \] (10)

Thus the indicator \( \psi_{k\ell} \) is always bounded between \([0, 1]\). At each local iteration \( \psi_{k\ell} \) is tested against a random threshold \( t \sim U(0, 1) \). If \( \psi_{k\ell} < t \) the current local search is terminated, yielding an exit probability at the \( k\ell \)-th local iteration of:

\[ p_{k\ell} = 1 - P(t \leq \psi_{k\ell}) = 1 - \psi_{k\ell}. \]

Intuitively, \( \psi_{k\ell} \) reflects our beliefs about the current centroid. A value of 0 implies that the current trust region centroid \( x_{k\ell} \) is a local minimum, and we should terminate the current trust region (local) search. Whereas a value of 1 reflects that a local minimum is not near \( x_{k\ell} \) and we should continue the current trust region search.

Given that a trust region (local) search is terminated and a local search restart is called for we return to a global search to re-estimate the global GP model and determine an optimal location for the local search restart. The new global GP estimation is executed over an updated set of observations, including all previous globally sampled points as well as all local points yielded by (7). Note that the \( 5 \times d \) initializing samples used for estimating local models are not included in this updated global set of samples. The intuition is that, at each local iteration, the observations from (7) are highly informative samples which and consolidate the information from each estimated local GP in a meaningful way, i.e., they are the result of maximizing the expected improvement sampling criteria over the locally estimated GP. Moreover, the inclusion of local expected improvement samples in the global GP allows the global model to embed learned local information and effectively characterize the locally modeled regions - while avoiding over fitting to any one local region.

Finally upon the update of the global GP, maximizing of the crowded expected improvement over the global GP yields a candidate local search restart centroid (9). Due to the symmetric nature of the hypercube trust region, a proposed candidate restart centroid is rejected if it lies on/near an input constraint, i.e., if any component of the candidate \( x_k \) is too near an upper or lower lower bound then it is rejected. If the candidate is rejected as a local search restart location it is sampled and appended to the global set of samples; the global GP is then updated/re-estimated (without any local search occurring) and an additional local search restart candidate is proposed.

IV. TEST CASES

We test 12 experimental cases over the three benchmark models, with six specifications over the first model.
and three specifications over each of the other two models, all models are implemented with Simulink/Stateflow models. All benchmarks and STL specifications used in this work are publicly available with download of the S-TaLiRo toolbox at https://sites.google.com/a/asu.edu/s-taliro/s-taliro

The falsification of an STL specification, φ, occurs when an output signal y ∈ Y produced by an input signal u ∈ U does not satisfy φ, where Y and U are the output and input spaces, respectively. From the robustness interpretation in [28], an observation map O : AP → P(Y) maps general atomic propositions p ∈ AP to a subset of Y. A system specification φ can be formalized mathematically as a temporal logic combination of atomic propositions p. Common semantics used in temporal logic include logical operators like negation (¬), conjunction (∧), disjunction (∨), and implies (→), as well as temporal operators like always (□), eventually (◇), next state (X), and until (U).

### A. Tests Description

The first benchmark problem is the Automatic Transmission (AT) example presented in [29], that is a modified version of the demo model provided as a MathWorks Simulink demo. This model has a single input of throttle schedule u(t) ∈ [0, 100] with the constant brake schedule of 0 throughout the 30 second simulation duration, t ∈ [0, 30]. The system has two outputs: engine speed in RPM ω(t) and vehicle speed in MPH ν(t), constituting the output signal y(t) = [ν(t), ω(t)]T. There are six STL formulas φAT i over the AT model. Formulas φAT 1 and φAT 3 are of the form:

\[
φ_{AT 1} = \neg(ϕ_{AT 1}^T \land ϕ_{AT 2}^T),
\]

where \( O(ϕ_{AT 1}^T) = [120, +\infty) \times \mathbb{R}, \) \( O(ϕ_{AT 2}^T) = \mathbb{R} \times [4500, +\infty], \) \( Z_1 = [0, 10] \) and \( Z_3 = [0.7, 5], \) i.e. the vehicle should not eventually both go above 120 MPH, while eventually going over 4,500 RPM (for the interval \( Z \)). The second STL has formula:

\[
φ_{AT 2} = \neg(ϕ_{AT 1}^T \land ϕ_{[0,10]}^T),
\]

where \( O(ϕ_{AT 2}^T) = [125, +\infty) \times \mathbb{R} \) states the vehicle should not eventually exceed 120 MPH while going above 125 MPH for \( t \in [0, 10]. \) Formulas 4, 5, and 6 are specified over the hybrid output space of the AT model, considering continuous state of the system and discrete location of the AT hybrid automata. These tests consider when the system eventually goes to all possible states {steady_state, upshifting, downshifting}, and then exceeds a speed threshold \( ν_i, \) for \( i = 4, 5, 6 \) the STL formulas are:

\[
φ_{AT i} = \neg(ϕ_{AT i}^T \land ϕ_{AT i}^T \land ϕ_{AT i}^T),
\]

with \( ϕ_{AT i}^T = \{steady\_state\} \times [ν_i, +\infty) \times \mathbb{R}, \) \( ϕ_{AT i}^T = \{upshifting\} \times [ν_i, +\infty) \times \mathbb{R}, \) and \( ϕ_{AT i}^T = \{downshifting\} \times [ν_i, +\infty) \times \mathbb{R}, \) where \( ν_4 = 79, ν_5 = 79.5, \) and \( ν_6 = 80. \)

The second benchmark problem is the 3rd order \( Δ - \Sigma \) modulator Simulink model, as described in [30]. The inputs to this system consist of an initial condition in the hypercube \([-0.1, 0.1]^3\) and a one dimensional input signal taking values in the set \([u_m, u_M] \). We provide three tests for this model, all using the specification, \( φ_{Δ - \Sigma}, \) that the state of the system should always remain within the hypercube \([-1, 1]^3\). The three tests are yielded by reducing the distance between \( u_m \) and \( u_M, \) restricting valid input signals. The three input ranges are: \([-0.45, 0.45], [-0.40, 0.40], \) and \([-0.35, 0.35]. \)

The final benchmark is the Navigation (NV) model from [31] that holds state variables of: grid sector, \( x_1 \) and \( x_2 \), velocity, and \( x_1 \) and \( x_2 \) positions. Three specifications are tested.

\[
φ_{NV 1} = (\neg(ϕ_{NV 1}^T \land ϕ_{NV 2}^T),
\]

where \( O(ϕ_{NV 1}^T) = \{4\} \times [3,2,3,8] × [0.2, 0.8] × \mathbb{R}^2, \) and \( O(ϕ_{NV 2}^T) = \{8\} \times [3,2,3,8] \times [1.2, 1.8] \times \mathbb{R}^2, \) which states that for \( t \in [0, 25] \) the system should not be in sector 4 with \( x \) and \( y \) velocities in \([3,2,3,8] \) and \([0,2,0.8] \) until it reaches sector 8 with velocities \([3,2,3,8] \) and \([1.2, 1.8]. \)

\[
φ_{NV 2} = □(p_{NV 1}^T \land □p_{NV 2}^T),
\]

where \( O(ϕ_{NV 1}^T) = \{10\} \times \{x \in \mathbb{R}^4 \mid x_1 \geq 1.05 \land x_2 \geq 2\}, \) and \( O(ϕ_{NV 2}^T) = \{5\} \times \{x \in \mathbb{R}^4 \mid x_1 \leq 1 \land x_2 \leq 1.95\}; \) states that if the system reaches sector 10 with position \( x_1 \geq 1.05 \) and \( x_2 \geq 2 \) then it should always not be in sector 5 with position \( x_1 \leq 1 \) and \( x_2 \leq 1.95. \)

\[
φ_{NV 3} = □(p_{NV 1}^T \land □p_{NV 2}^T),
\]

where \( O(ϕ_{NV 1}^T) = \{10\} \times \{x \in \mathbb{R}^4 \mid x_1 \geq 1.2 \land x_2 \geq 2\}, \) and \( O(ϕ_{NV 2}^T) = \{5\} \times \{x \in \mathbb{R}^4 \mid x_1 \leq 1 \land x_2 \leq 1.9\}; \) states if the system has position \( x_1 \geq 1.2 \) and \( x_2 \geq 2 \) then it should always go to a position \( x_1 \leq 1 \) and \( x_2 \leq 1.9. \)

For benchmarking, all 12 test case are parameterized identically to [10]. All AT specifications have seven control points placed on the throttle signal \( u(t), \) which corresponds to a seven dimensional \( x \) in [1] and \( f(x) \) is robustness of trajectory \( x \) in the AT model with respect to specification \( φ_{AT i}^T \), and subsequently \( χ(φ_{AT i}^T) \subseteq \mathbb{R}^7. \) All modulator model specifications have 10 input signal control points and 3 initial conditions, \( χ(φ_{Δ - \Sigma}) \subseteq \mathbb{R}^3. \) All NV specifications have four initial state conditions as input, \( χ(φ_{NV i}^T) \subseteq \mathbb{R}^4. \)

### B. Results Discussion

We implement the SOAR optimization algorithm (outlined in Section [III]) in the S-TaLiRo toolbox in Matlab. To compare the effectiveness of SOAR in the S-TaLiRo tool we make use of a uniform random (UR) sample generator, as well as a Monte-Carlo based Simulated Annealing (SA) algorithm, both of which are available in the S-TaLiRo toolbox and compatible with hybrid robustness metrics. Note that the implemented SA algorithm holds special Monte-Carlo sampling properties from a dual Markov Chain formulation—such that the technique can be executed directly over hybrid output space [10]. Whereas our implemented SOAR
algorithm makes use an inverse logistic transform to relate observed hybrid output robustnesses measures into scalars capable of being modeled by a standard GP.

We present the results over the 12 test cases from Section IV-A in Tables III and IV. Table III reports results for model/specification combinations that yield Euclidean output spaces, while those in Table IV yield a hybrid output space consisting of a discrete location component and a continuous dynamics component. Each of the 12 experiments are run 100 times to account for the randomized nature of the algorithms, and each of the runs are executed with a budget of 1,000 tests (model simulations) per run.

The additional advantage of our implemented SOAR framework is that it produces a predictive meta-model of the robustness landscape, which can be leveraged as a means for assessing the safety of unfalsified systems. Reported in each table, the first three columns correspond to the number of falsifications each of the three sampling techniques are able to produce (out of 100 runs). The last two columns address those runs which were not falsified by SOAR. We present SOAR’s 95% confidence intervals for the predicted minimums, yielded from the final global GP models over those runs which were not falsified.

In Table III, the automatic transmission model shows three distinct behaviors of SOAR with respect to SA and UR. The first specification \( \phi_{AT}^1 \) is easily falsified and this is reflected by all three algorithms, as well as the strongly negative minimum prediction interval for the two SOAR runs that did not falsify. The second specification \( \phi_{AT}^2 \) shows a case where SA outperforms both competitors; however, SOAR is able to produce one falsifying trajectory whereas UR is unable to produce any. The last specification \( \phi_{AT}^6 \) is a difficult falsification problem, where no competitors could produce a falsification, this is a key case where, despite a lack of finding any falsifications, SOAR’s final prediction model consistently produces a minimum prediction that falsifies. As a practitioner such a result gives insight to the need for further testing and that, though no unsafe behavior has been observed, unsafe behaviors likely exist. This is a distinct case where the additional information from SOAR provides insight into system behavior that neither SA nor UR can produce.

The hybrid output specifications in Table IV show a bit more difficulty for the application of our implemented SOAR algorithm. It is of note that SA performs well over both \( \phi_{AT}^4 \) and \( \phi_{AT}^6 \), illustrating the effectiveness of utilizing the parallel dual Markov Chain approach over this hybrid output space and lending insight into the best ways to approach these hybrid output problems. However, over both of these specifications we still see that the final models produced by SOAR (produced with the inverse logistic transformed output space) yield predictions that an unsafe trajectory likely exists. Moreover, the specification \( \phi_{AT}^6 \) presents another case where it seems the system may be safe, indeed SA-a strong performer over this model- is unable to find any falsifications. The final GP models produced by SOAR agree with this lack of falsifications and predict that it is unlikely that a falsification exists, an additional piece of information for practitioners to pair with the lack of finding any falsifying inputs. Specification \( \phi_{AT}^{NV} \) of the navigation model gives insight into understanding potential robustness landscape/sampling interactions. Clearly SOAR is able to regularly find falsifying signals, however, those runs which are unable to produce falsifications predict that it is unlikely for a falsification to exist. In such cases it is possible for there to exist regions of the solution space with substantially different robustness behavior, where those runs which do not falsify fit global models to regions where the behavior is consistently strong and shows no signs of deterioration with respect to the specification in question.

From Table IV it seems that there is room for further research into applying more specialized techniques for the generation of meta-models. These meta-models could directly address the issue of the hybrid output spaces by

<table>
<thead>
<tr>
<th>Problem</th>
<th># Falsifications</th>
<th>Pred. Min. CI</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT</td>
<td>( \phi_{AT}^1 )</td>
<td>[98, 97, 100]</td>
</tr>
<tr>
<td>AT</td>
<td>( \phi_{AT}^2 )</td>
<td>[1, 52, 0]</td>
</tr>
<tr>
<td>AT</td>
<td>( \phi_{AT}^6 )</td>
<td>[0, 0, 0]</td>
</tr>
<tr>
<td>NV</td>
<td>( \phi_{NV}^1 )</td>
<td>[88, 63, 84]</td>
</tr>
<tr>
<td>NV</td>
<td>( \phi_{NV}^2 )</td>
<td>[77, 37, 44]</td>
</tr>
<tr>
<td>NV</td>
<td>( \phi_{NV}^6 )</td>
<td>[24, 19, 0]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Problem</th>
<th># Falsifications</th>
<th>Pred. Min. CI</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT</td>
<td>( \phi_{AT}^1 )</td>
<td>[67, 97, 85]</td>
</tr>
<tr>
<td>AT</td>
<td>( \phi_{AT}^2 )</td>
<td>[55, 97, 59]</td>
</tr>
<tr>
<td>AT</td>
<td>( \phi_{AT}^6 )</td>
<td>[0, 0, 0]</td>
</tr>
<tr>
<td>NV</td>
<td>( \phi_{NV}^1 )</td>
<td>[72, 68, 32]</td>
</tr>
<tr>
<td>NV</td>
<td>( \phi_{NV}^2 )</td>
<td>[100, 100, 100]</td>
</tr>
<tr>
<td>NV</td>
<td>( \phi_{NV}^6 )</td>
<td>[11, 34, 1]</td>
</tr>
</tbody>
</table>
modeling directly over the hybrid space (as SA does). Additionally there seems to be potential for multiple meta-models to iteratively identify and fit regions of the solution space with substantially differing behavior. These multiple models could then be composed into a much more refined global model that is a mixture of several models, with each model accurately capturing the behavior of a region with specific robustness landscape characteristics.

V. CONCLUSIONS

We address the problem of cyber-physical system falsification with respect to STL system specifications with limited testing budget, and propose the use of the SOAR framework for test case generation. We detail a fully specified version of the SOAR framework in Section III which makes use of a single global Gaussian process (GP) model as well as a trust region based local search which leverages local GP models to exploit regional characteristics of the robustness landscape.

Our specified SOAR algorithm is implemented in the STaLiRo tool in Matlab and three benchmark models with several system specifications are tested in Section IV. SOAR outperforms both Simulated Annealing and uniform random sampling in the benchmark testing, and provides confidence levels over the predicted minimum robustness value when no falsifications can be found. A characteristic that no other falsification tools currently have. This additional piece of information can give insight to practitioners as to the degree of further testing that is needed when no falsifying inputs are found.

REFERENCES