

Preventive Controlled Islanding Against Multiple Probabilistic Cyber Threats in Power Grids

Sho Cremers^{*†}, Alexandru Ștefanov[†], Marten van Dijk^{*‡}, Han La Poutré^{*†}

^{*}*Centrum Wiskunde & Informatica (CWI), Amsterdam, The Netherlands*

[†]*Delft University of Technology, Delft, The Netherlands*

[‡]*Vrije Universiteit Amsterdam, Amsterdam, The Netherlands*

{sho.cremers, marten.van.dijk, han.la.poutré}@cwi.nl, a.i.stefanov@tudelft.nl

Abstract—While digitalization and integration of communication networks play a vital role in modern power system operation, they also expose the power grid to potential cyber attacks that can cause significant societal disruptions. Considerable effort is devoted to the early detection of cyber intrusions. However, the detection of an adversary does not guarantee that attacks can be neutralized before physical disturbances occur. This work proposes a preventive controlled islanding method to enhance the power grid resilience against multiple anticipated attacks with known probabilities. The proposed approach is formulated as a graph-constrained coalition formation problem, focusing on grouping electrical nodes rather than directly selecting lines to disconnect, and determining islanding solutions using exact and adapted Monte Carlo tree search (MCTS) algorithms. Case studies demonstrate that both the locations and probabilities of attacks influence the optimal islanding strategy. Furthermore, the proposed formulation allows the optimal number of islands to emerge from the solution rather than being a preset parameter. The results also show that MCTS can identify (near-)optimal solutions with substantially less computation time than the exact algorithm.

Index Terms—cyber attack, grid resilience, preventive controlled islanding, coalition formation

I. INTRODUCTION

While digitalization and communication infrastructures have become critical for the monitoring, control, and operation of modern power systems, they have also introduced exposures to cyber security threats. Network-connected devices in the field and remote access capabilities to operational technology (OT) systems increase the risk of unauthorized access and control over operations. Cyber attacks on power grids may cause power outages with critical consequences for public well-being and society. Notable examples include the attacks on the Ukrainian power grid in 2015 [1] and 2016 [2], which left many consumers without power. Thus, cyber security and resilience enhancement are recognized as critical requirements for power grid operation.

Intrusion detection systems (IDSs) play an important role in the early detection of cyber intrusions in power systems, with an increasing number of studies focusing more on data-driven anomaly-based detection compared to traditional signature-based detection [3]. Such IDSs deployed locally on substation

communication networks may provide early detection. Similarly, a centralized approach that inspects OT communication network traffic has been shown to detect and localize intrusions with high accuracy [4]. Furthermore, many anomaly-based IDSs produce a continuous anomaly score that quantifies the deviation of an observation from expected behavior. In addition to binary detection decisions derived by applying a threshold to the anomaly score, these scores can be interpreted or calibrated to provide confidence measures regarding the likelihood of a compromise. We interpret the confidence level as the attack probability. The localization of the intrusion, along with the confidence level, can enable more informed decision-making during incident response.

Using information about potential cyber intrusions and attacks in the physical network, we propose a preventive controlled islanding framework to enhance power grid resilience. Controlled islanding has been extensively studied as a response strategy *after* physical disturbances, in which the power grid is intentionally separated into multiple islands to improve system stability and avoid cascading failures [5]. In contrast, preventive controlled islanding partitions the power grid *before* any physical disturbance occurs and has been studied under extreme weather conditions [6], where the affected area can often be anticipated. Furthermore, preventive islanding has recently been identified as a promising approach for enhancing power grid resilience against cyber attacks and mitigating the risk of blackout, as part of [7]. Unlike weather-related disturbances, cyber attacks may occur simultaneously at multiple locations and generally cannot be predicted in advance. Cyber security professionals aim to detect and mitigate attacks in cyberspace before they impact power system operation. A coordinated cyber-physical incident response is needed to anticipate and prepare the power grid to withstand the attack impact and avoid a blackout. Based on IDS alerts, power grid operators may compute an islanding solution that is implemented before the attacks result in opening circuit breakers to prevent cascading failures. Preventive islanding may also reduce load loss due to a large number of compromised third-party devices, such as smart home appliances and EV charging stations. Although intrusion information may be available, the system operator has limited control over the cyber security of these devices and assets, motivating a resilience-oriented response.

In this work, we present a mathematical model of con-

trolled islanding under multiple probabilistic attack scenarios. We extend this formulation to a graph-constrained coalition formation (GCCF) framework [8]. With this formulation, electrical buses are seen as agents that can form coalitions (or islands) subject to connectivity constraints imposed by the physical power grid topology. The buses in the power grid are partitioned into exhaustive and disjoint coalitions, forming a coalition structure (CS), with each coalition evaluated using a coalition value function. The objective is then to identify a CS that maximizes social welfare (or serviced load), which, in this work, corresponds to minimizing the potential physical disruption caused by an attacker.

The advantages of modeling controlled islanding as GCCF are as follows. First, the coalition value function can be defined flexibly while keeping the overall optimization problem unchanged. For example, if the islands' values are derived from optimal power flow (OPF), then the function is interchangeable across different OPF models, depending on the user's requirements and performance trade-offs. The second advantage is that the utility of each island can be evaluated independently of the other islands. Third, and importantly, the optimal number of islands arises naturally from the identified solution, contrary to the commonly used graph clustering algorithms, which require the number of islands as a parameter or an additional step to determine the number [5]. This property of GCCF makes it especially suitable for a cyber attack scenario, where multiple points in the power grid may be compromised, and hence requires multiple islands.

The contributions of this work are as follows.

- First, we propose a mathematical framework and method for preventive controlled islanding to enhance the power grid resilience to cyber attacks, when the attack locations and attack probabilities are assumed to be known. We formulate the problem as a GCCF, in which the advantages of such a formulation are noted above.
- Second, exact and Monte Carlo tree search (MCTS) algorithms for GCCF are used and adapted to determine the islanding solutions.
- Third, we evaluate the proposed approach on the IEEE 14-bus system [9] by considering cyber intrusions targeting communication networks associated with electrical buses. The results demonstrate that both the locations and probabilities of attacks influence the optimal islanding strategy, including the varying number of islands. Furthermore, MCTS is able to find (near-)optimal solutions while requiring less computation time than the exact algorithm.

The remainder of this paper is organized as follows. Section II reviews the related literature. Section III presents the proposed GCCF-based formulation for preventive islanding with multiple probabilistic attacks. Section IV describes the solution approaches. Section V presents the case studies and experimental results. Finally, Section VI concludes the paper.

II. RELATED WORK

A. Controlled Islanding Against Malicious Attacks

Controlled islanding strategies against cyber attacks have previously been studied. [10] modeled the interaction between an attacker capable of disconnecting transmission lines through a cyber intrusion and a system operator performing islanding as a repeated zero-sum game. [11] considered cyber attacks on substations that result in the disconnection of multiple transmission lines and formulated the interaction between the attacker and the operator as a Stackelberg game. [12] considered controlled islanding under uncertainty caused by false data injection attacks, focusing on creating observable and stable islands while isolating potentially compromised measurements. [13] proposed a near-real-time controlled islanding strategy based on spectral clustering to mitigate cascading failures following cyber attacks on transmission lines and generators. While these works focus on mitigation actions during or after a disturbance, [7] identified preventive islanding as one of several promising approaches to cyber threats. Our work complements this perspective by providing a mathematical framework for preventive islanding, extending it to multiple probabilistic attack scenarios, framing it as coalition formation, and quantitatively evaluating its effectiveness.

B. Graph-Constrained Coalition Formation in Energy

Coalition formation over a graph has gathered attention for applications to energy systems. [14] studied the formation of groups that balance power generation and demand using small and tractable transmission networks as case studies. Although connectivity constraints were not considered, DC power flow constraints were incorporated into their coalition value function. More recently, [15] and [16] modeled energy community formation in distribution networks as a graph-constrained coalition formation (GCCF) problem. Their studies abstracted the problem as an induced subgraph game and solved it using quantum annealing-based algorithms. [8] proposed several algorithms for solving GCCF, with collective energy purchasing as one of their case studies. They introduced an exact algorithm that evaluates each valid CS exactly once, as well as branch-and-bound algorithms that exploit specific properties of the coalition value function to improve efficiency.

Note that these works focus on economic coordination rather than operational resilience of the system. Furthermore, with the exception of the exact algorithm in [8], the proposed solution methods require the coalition value function to satisfy certain properties or rely on an abstracted representation of the problem. In contrast, the Monte Carlo tree search (MCTS) algorithm for GCCF proposed in [17] can determine near-optimal solutions efficiently, while making no assumptions about the value function. Our work uses and adapts [8] and [17] for preventive controlled islanding.

III. PROBLEM FORMULATION

A. GCCF Formulation for Controlled Islanding

The topology of the physical network is represented as an undirected graph $G = (V, E)$, with the set of nodes V for elec-

trical buses and the set of edges E for branches (power lines and transformers). While this work focuses on transmission networks with a mesh topology and all lines E active during normal operation, the formulation can be applied to different topologies and line statuses. Let us denote a subgraph of G induced by a set of nodes $N \subseteq V$ as $G_N = (N, E_N) \subseteq G$, where E_N is the set of all edges that connect pairs of nodes in N , i.e., $E_N = \{(x, y) \in E | x, y \in N\}$. Henceforth, N denotes a coalition of nodes, and the subgraph G_N is the corresponding island in the physical network induced by N .

We define the coalition value function $c(N)$ to quantify the quality of coalition N . In our work, $c(N)$ represents the load served by the corresponding island G_N (see Section III-B). Although the evaluation of an island is based on the induced subgraph $G_N = (N, E_N)$, we use the notation $c(N)$ to remain consistent with the coalitional game theory literature.

The overall optimization problem can be modeled as maximizing the sum of coalition values, or the total utility:

$$\max_{\pi \in \mathfrak{P}(V)} \sum_{N \in \pi} c(N), \quad (1)$$

where π is a set of exhaustive and disjoint coalitions called a coalition structure (CS), i.e.,

$$\bigcup_{N \in \pi} N = V, \text{ and } [N_i \cap N_j = \emptyset \quad \forall N_i, N_j \in \pi, i \neq j]. \quad (2)$$

The set $\mathfrak{P}(V)$ denotes all CSs that can be formed by V . Note that in GCCF, coalitions are constrained by their physical properties, i.e., G_N is a connected subgraph for every $N \in \pi$.

B. Preventive Controlled Islanding Against Cyber Intrusion

In this work, each node $v \in V$ is assigned an attack probability $p_v \in [0, 1]$. We assume that an attacker can intrude into multiple nodes, and their probabilities are independent. If nothing is done while the attacker is present, the connected part will experience a blackout due to the attacker spreading through the communication network and cyber-physical attacks that cause cascading failures. Hence, preventive islanding aims to isolate threatened areas while continuing to meet power demand where feasible. Note that the network partitioning by the system operator not only separates the physical network but also assumes it blocks communication between nodes from different islands; hence, the attacker's control cannot spread beyond the island it is contained in.

The value function of a coalition N represents the *expected weighted served load* of the induced subgraph G_N , denoted as a product of two functions:

$$c(N) = g(N)f(N). \quad (3)$$

The first function represents the probability that the island is *not* subject to failure caused by cyber attacks:

$$g(N) = \prod_{v \in N} (1 - p_v). \quad (4)$$

The second function $f(N)$ is the weighted served load via the optimal power flow (OPF) given the subgraph $G_N = (N, E_N)$:

$$f(N) = \max_{\mathbb{P}(G_N)} \sum_{v \in N} \lambda_v \hat{P}_v^l, \quad (5)$$

where $\mathbb{P}(G_N)$ represents the set of constraints, parameters, and variables of the AC-OPF formulation for network G_N , as implemented in [18] based on the formulation of [19]. The weight parameter λ_v quantifies the importance and priorities of the load in node v . The variable $\hat{P}_v^l$ represents the served active power load at node v . In this work, generation curtailment and load shedding are also considered in the event of load-generation imbalances when islands form. Thus, the generator constraints in $\mathbb{P}(G_N)$ bound the active power output of each generator between 0 and its pre-islanding value. Additionally, load shedding at node v is constrained by a shedding factor $\alpha_v \in [0, 1]$ which maintains the same active and reactive power shedding ratio [20]. Therefore, the following load-shedding constraints are also included in $\mathbb{P}(G_N)$:

$$\hat{P}_v^l = \alpha_v P_v^l, \quad \hat{Q}_v^l = \alpha_v Q_v^l, \quad \forall v \in N, \quad (6)$$

where P_v^l and Q_v^l are the active and reactive power demands that are met during normal operation. The variables $\hat{P}_v^l$ and $\hat{Q}_v^l$ denote the served active and reactive power load at v after the island G_N is formed. Note that, while the AC-OPF model is employed in this work, the proposed framework is not tied to a specific operational model. The underlying constraints of the value functions may be relaxed to improve computational efficiency or extended to incorporate additional operational considerations, according to the application requirements.

Essentially, if there is no attacker ($p_v = 0, \forall v \in V$), the grand coalition (a fully connected network G) is optimal, as the system operator makes sure the network is operated safely and the generation and load are balanced, i.e., $c(V)$ is a solution to (1). If some v has $p_v = 1$, it may be trivial to form an independent coalition solely by v . However, if $0 < p_v < 1$, neighboring nodes may join v to form a coalition that increases the overall expected load instead of other coalitions with no attack probability, as the served load increases more than the expected loss due to the failure of the nodes in the coalition.

IV. SOLUTION METHODS

A. Exact Algorithm

All valid CSs in GCCF can be determined by repeated edge contractions, each representing the merging of coalitions in the graph. [8] showed that a search tree that contains all feasible CSs in a graph exactly once can be constructed by representing a graph-constrained CS as a two-colored graph, where the edges can be marked as green or red. In this section, the exact algorithm from [8] is described, where it is reformulated using a search tree, subsequently applied in MCTS.

Let us define the search tree as $T = (S, A)$ where S is the set of tree states, and A is the set of tree transitions or actions. While S and A represent the tree nodes and edges, the terms *states* and *actions* are used to differentiate them from the graph nodes V and edges E . A tree state $s \in S$ represents

the CS state of the network buses and the colored hyperedges (between coalitions), which is defined as a tuple:

$$s = (\pi_s, H_s^G, H_s^R), \quad (7)$$

where π_s is the graph-constrained CS of the electrical network (see (1)). H_s^G and H_s^R are the sets of green and red hyperedges of the network, respectively, that link two coalitions in π_s . A hyperedge h may either be a single graph edge or may consist of several edges in E if multiple edges exist between the two coalitions. Green hyperedges of a state represent links between coalitions that can be merged in its child states, whereas red hyperedges are links between coalitions that cannot be merged. Keeping track of red hyperedges avoids generating duplicate CSs in the search tree, and hence prevents unnecessary utility computations.

The root state of the search tree is the CS in which each node in the graph is an independent coalition, and all edges in E are set as green hyperedges, defined as:

$$s_{root} = (\{\{v\} | v \in V\}, E, \emptyset). \quad (8)$$

A tree action $a \in A$ represents contraction of a hyperedge in H_s^G at state s . Additionally, some green hyperedges in state s are converted to red. An action is defined as follows:

$$a = (h, H^\Delta), \quad (9)$$

where $h \in H_s^G$ is the contracting green hyperedge and $H^\Delta \subseteq (H_s^G \setminus \{h\})$ is the set of hyperedges to be converted to red. The set H^Δ is determined by the algorithm and is used to avoid duplicate CSs in the search tree. Each time a child state is generated by contracting h in the exact algorithm, the hyperedge is added to H^Δ , thereby ensuring that subsequent sibling states treat it as red (see Algorithm 1).

The child state in the search tree can be defined as a function of the parent state s and action a :

$$s_{ch} = \text{NEXTSTATE}(s, a) = (\pi_{ch}, H_{ch}^G, H_{ch}^R). \quad (10)$$

Hyperedge contraction merges two coalitions. Given the parent state s and state action $a = (h, H^\Delta)$ with $h = (N_i, N_j)$ that links the coalitions $N_i, N_j \in \pi_s$, the new CS after contraction can be denoted as follows:

$$\pi_{ch} = (\pi_s \setminus \{N_i, N_j\}) \cup \{N_i \cup N_j\}. \quad (11)$$

In addition to the CS, the sets of green and red hyperedges are also updated at the child state. First, H^Δ is removed from the green hyperedges ($H_s^G \setminus (H^\Delta \cup \{h\})$), whereas the same hyperedges are added to the set of red hyperedges ($H_s^R \cup H^\Delta$). Second, because H_s^G and H_s^R represent links between coalitions rather than nodes in the graph, they need to be updated after the contraction. This includes combining green hyperedges between the same coalitions and maintaining a single hyperedge between the coalitions. Similarly, only a single red hyperedge between the two coalitions is kept, in case there is more than one red hyperedge between two coalitions. Additionally, a red hyperedge is considered dominant; if green and red hyperedges both exist between two coalitions, the

resulting hyperedge is red. For readability reasons, let us denote the new green and red hyperedges as a function of parent state s and action a :

$$H_{ch}^G = \text{UPDATEGREENEDGES}(s, a), \quad (12)$$

$$H_{ch}^R = \text{UPDATEREDEDGES}(s, a). \quad (13)$$

From (11) – (13), the child state as in (10) can be generated. Given the definition of the search tree state and action, the exact algorithm determines the optimal graph-constrained CS by exhaustively visiting the tree states and evaluating unique CSs. The entire procedure is summarized in Algorithm 1.

Algorithm 1 SOLVEGCCF(G, s_{root})

```

 $\pi_{best} \leftarrow \pi_{root}, F \leftarrow \{s_{root}\}$ 
while  $F \neq \emptyset$  do
   $s \leftarrow F.\text{POP}()$ 
  if  $\sum_{N \in \pi_s} c(N) > \sum_{N \in \pi_{best}} c(N)$  then
     $\pi_{best} \leftarrow \pi_s$ 
     $H^\Delta \leftarrow \emptyset$ 
    for  $h \in H_s^G$  do
       $F.\text{PUSH}(\text{NEXTSTATE}(s, (h, H^\Delta)))$ 
       $H^\Delta \leftarrow H^\Delta \cup \{h\}$ 
return  $\pi_{best}$ 

```

Figure 1 shows a search tree of an example graph after the children of the root state are generated according to the exact algorithm. Here, the children are generated in the order of the actions $\{a_1, a_2, a_3, a_4\}$, and it can be seen that H^Δ grows with the subsequent actions. Note that states s_1, s_2 , and s_3 still contain green hyperedges, and hence they can further generate new states, while s_4 is a terminal state in the search tree.

B. Monte Carlo Tree Search

[17] proposed an MCTS method that efficiently searches the graph-constrained CSs without making any assumptions on the value function $c(\cdot)$. In this section, the MCTS algorithm for preventive islanding is described and adapted using the previously defined search tree formulation $T = (S, A)$. Each state s in the constructed search tree of MCTS maintains, in addition to the variables in (7), the value u_s and CS

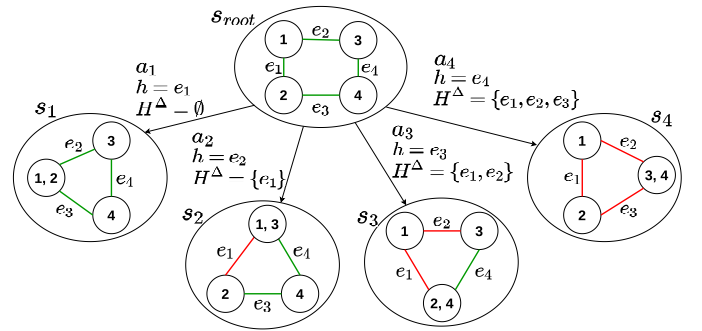


Fig. 1. Child states generated from the root state in the search tree.

$\pi_s^{\max}$, representing the highest utility and the corresponding CS observed through s and some of its descendants; the count n_s , representing the number of traversals through s ; and a set $H_s^C \subseteq H_s^G$, representing the green hyperedges that have previously been contracted from s , initialized as $H_s^C \leftarrow \emptyset$.

Algorithm 2 MCTSGCCF(G, s_{root}, m)

```

 $T = (\{s_{root}\}, \emptyset)$  ▷ initialize search tree
for  $i = 1$  to  $m$  do ▷ iterate through  $m$  samples
   $s \leftarrow \text{SELECTION}(T)$  ▷ traverse search tree
  if  $H_s^G \setminus H_s^C \neq \emptyset$  then ▷ state not yet fully expanded
     $(s, T) \leftarrow \text{EXPANSION}(s, T)$  ▷ add new state to tree
     $(u'_s, \pi'_s) \leftarrow \text{SIMULATION}(s)$  ▷ estimate max utility
     $\text{BACKPROPAGATION}(u'_s, \pi'_s, s, T)$  ▷ update ancestors
return  $\pi_{s_{root}}^{\max}$  ▷ CS with max utility found

```

The overall procedure of the MCTS algorithm for GCCF is shown in Algorithm 2. Initially, the tree contains only the root state s_{root} . For a preset number of samples m , it repeats the following steps. First, in the *selection* step, it traverses the constructed search tree starting from the root state to select a state s . Next, in the *expansion* step, if s is not yet fully expanded, i.e., not all child states of s are added to the search tree, then one of its child states is added to the tree, and s is updated to the newly added state. Next, the *simulation* step estimates the maximum utility attainable from s and its descendants by computing the utilities of CSs that can be formed by successively contracting its green hyperedges H_s^G . Finally, during *backpropagation*, the search tree is traversed upwards from s to the root and updated, such that each state keeps track of the best utility and the corresponding CS of its processed descendants from previous samples. After m samples, the best CS stored at the root state is returned. The four main steps of MCTS are explained in more detail below.

1) *Selection*: Starting from the root state s_{root} , the search tree is traversed by choosing its child state s_{ch} with the highest UCT score, which balances exploitation and exploration using $u_{s_{ch}}$ and $n_{s_{ch}}$ (see [17]). Hence, the selected state s is updated as $s \leftarrow \arg \max_{s_{ch} \in \text{Children}(s)} \text{UCT}(s_{ch})$, where $\text{Children}(s)$ represents all child states of s that can be generated by contracting one of the green hyperedges in H_s^G , and $\text{UCT}(s)$ is the UCT score of s . The procedure is repeated recursively until one of the following conditions is met: (i) s is a terminal state, i.e., $H_s^G = \emptyset$; or (ii) s contains at least one hyperedge $h \in H_s^G$ that has not been contracted in previous samples, i.e., $H_s^G \setminus H_s^C \neq \emptyset$. In the latter case, the following step is applied.

2) *Expansion*: In this step, the search tree is expanded by adding a new state if the selected state s (from the previous step) is not fully expanded, i.e., it has a child that has not yet been added to the search tree. Given s , a hyperedge h is randomly chosen from the set $(H_s^G \setminus H_s^C)$. Then the child state s_{ch} is generated as $s_{ch} = \text{NEXTSTATE}(s, (h, H_s^C))$, and is added to search tree T . Subsequently, the set of already contracted hyperedges of s is updated as $H_s^C \leftarrow H_s^C \cup \{h\}$, preventing the same hyperedge from being contracted again

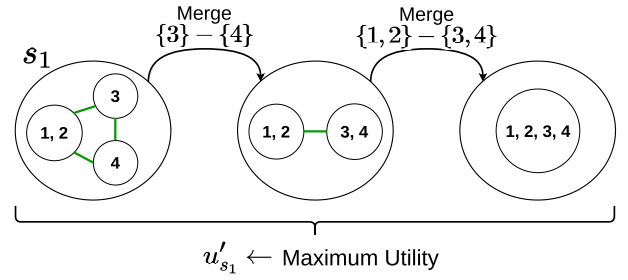


Fig. 2. Simulation step from MCTS at s_1 of Fig. 1.

when s is encountered during tree traversal of the selection step in future samples. The set H_s^C therefore serves the same purpose as H^Δ in the exact algorithm (see Fig. 1). Unlike the exact algorithm, however, MCTS does not generate all child states of s simultaneously. Instead, child states are expanded incrementally across different samples, requiring H_s^C to be stored for every state in the search tree. Finally, the selected state is updated before moving on to the next step: $s \leftarrow s_{ch}$.

3) *Simulation*: This step estimates the maximum utility attainable from s . Following [17], CSs are generated by successively contracting green hyperedges of s . Let $R(s) = \{\pi^0, \pi^1, \dots, \pi^k\}$ denote the set of CSs encountered during this contraction process, where $\pi^0 = \pi_s$ and each subsequent CS is obtained by contracting the remaining green hyperedge with the largest weight. The process ends when no green hyperedges remain. Then, the best CS and its utility from $R(s)$ are determined:

$$\pi'_s \leftarrow \arg \max_{\pi \in R(s)} \sum_{N \in \pi} c(N) \quad \text{and} \quad u'_s \leftarrow \sum_{N \in \pi'_s} c(N). \quad (14)$$

An example of the simulation step of state s_1 from Fig. 1 can be seen in Fig. 2. Here, $R(s_1)$ consists of CSs $\{\{1, 2\}, \{3\}, \{4\}\}$, $\{\{1, 2\}, \{3, 4\}\}$, and $\{\{1, 2, 3, 4\}\}$, and u'_{s_1} is assigned the maximum utility of the three CSs.

As a contribution of this work and adaptation of MCTS to preventive islanding, we determine the hyperedge weights such that the generation surplus/deficit is taken into account for ordering of hyperedge contractions. The weight is assigned as the negative product of the active power surplus (independent of operational constraints) of the two coalitions, i.e., the weight of a hyperedge h_{ij} between coalitions N_i and N_j is given as follows:

$$w_{ij} = - \left(\sum_{v \in N_i} (P_v^g - P_v^l) \right) \cdot \left(\sum_{v \in N_j} (P_v^g - P_v^l) \right), \quad (15)$$

where P_v^g denotes the active power generation at v during normal operation (before islanding). The main reasons for using such weights are the following. First, the weight is positive only if one coalition has a generation surplus and one has a generation deficit. Second, the weight is higher when both coalitions have a large imbalance value, indicating greater potential to improve the load-generation balance if the coalitions are merged.

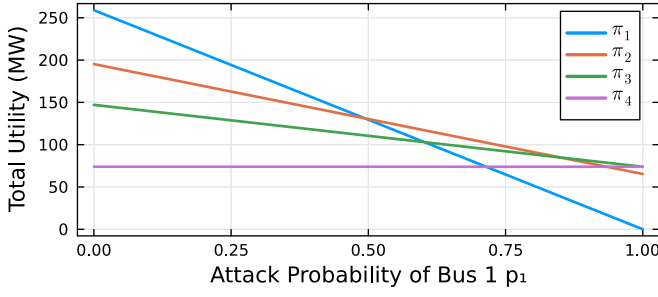


Fig. 3. Utilities of four CSs for varying p_1 . The grand coalition (fully connected network) is denoted by π_1 .

4) *Backpropagation*: After the simulation step, if the value u'_s exceeds the assigned value u_s , then u_s and $\pi_s^{\max}$ are updated accordingly. Additionally, the count n_s is incremented by one. The same update is propagated recursively to each ancestor state up to s_{root} . Consequently, u_s represents the highest utility observed so far in the subtree rooted at s , and thus a lower bound on the maximum utility within that subtree. In particular, $u_{s_{root}}$ always contains the highest utility found during the search.

V. CASE STUDIES

The 14-bus system [9], as shown in Fig. 4, was considered to examine the impact of preventive controlled islanding. All loads were set to equal weights ($\lambda_v = 1$), and the attack probabilities p_v were set to 0 unless otherwise specified. Experiments were conducted on a laptop equipped with an Apple M1 processor and 16 GB RAM, with the experimental code implemented in Julia (v1.12.2). The AC-OPF formulation for $c(\cdot)$ was modeled with `PowerModels.jl` [18] (v0.21.5), and solved using the default IPOPT solver (v1.13.0).

A. Impact of Attack Probability

Here, bus 1 in the network was the target of a cyber attack. Bus 1 plays a crucial role as it is connected to a generator supplying about 80% of the total active power load. Using the exact algorithm as shown in Section IV-A, the optimal islands and their utilities were evaluated for varying probabilities p_1 .

Figure 3 shows utilities of the four CSs that are optimal for some p_1 as the attack probability increases, computed by steps of 0.001. The corresponding CSs are listed in Table I. The y-intercepts (values when $p_1 = 0$) of the lines in Fig. 3 capture the utility achieved in the absence of an attack and hence reflect the preventive islanding trade-off prior to any disruption. The slopes can be interpreted as the damping of

TABLE I
CSs THAT ARE OPTIMAL FOR CERTAIN p_1 VALUES.

Name	# Coalitions	Coalitions
π_1	1	{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14}
π_2	3	{1, 4, 5, 6, 7, 8, 9, 11, 12, 13, 14}, {2, 3}, {10}
π_3	2	{1, 5, 6, 9, 10, 11, 12, 13, 14}, {2, 3, 4, 7, 8}
π_4	4	{1}, {2, 3, 4, 5, 7, 8}, {6, 10, 11, 12}, {9, 13, 14}

TABLE II
ATTACKS ON BUS 1 WITH $p_1 = 0.9$.

Methods	Utility (MW)				Time (s)	
	Mean	SD	Max	Min	Mean	SD
MCTS-50	73.99	4.67	80.79	65.41	0.76	0.11
MCTS-500	76.89	2.81	80.91	72.19	1.80	0.18
MCTS-5000	79.52	1.84	81.22	76.14	4.15	0.39
Exact	81.22	—	—	—	20.26	0.15

the attack impact that the islanding strategy has. Trivially, all CSs show non-increasing utility as p_1 increases, as failure of the coalition containing bus 1 increases with p_1 (from (4)). The CS π_1 represents a fully connected network, and remains the optimal solution until $p_1 \approx 0.5$. For larger p_1 , other CSs achieve higher utilities, indicating that islanding can reduce potential losses when the likelihood of attacks is sufficiently high. Moreover, different CSs become optimal at different values of p_1 , showing that the attack probabilities influence the optimal islanding strategy.

Table I also shows that the coalition containing bus 1 decreases in size (number of buses) as p_1 increases, reflecting the gradual isolation of a potentially compromised bus. Furthermore, the optimal number of coalitions can vary with p_1 . This emphasizes the advantage of formulating controlled islanding as a GCCF problem rather than a graph-cut problem: the number of islands emerges as part of the optimization process, whereas graph clustering-based approaches typically require selecting the number of islands in advance.

B. Algorithm Performance Comparison

Here, the utility and computation time of the MCTS algorithm were compared to the exact algorithm as the benchmark. Given sufficient samples, MCTS explores the entire search tree and thus finds the optimal solution, though less efficiently than the exact algorithm. To examine the trade-off between the computation time and utility of MCTS, sample sizes of 50, 500, and 5000 were considered. Table II presents the results when bus 1 is attacked with $p_1 = 0.9$. The table reports the mean utility over 20 independent runs and its standard deviation, along with the maximum and minimum values observed. The mean computation time and its standard deviation are also provided. The mean utility of MCTS gradually increases with

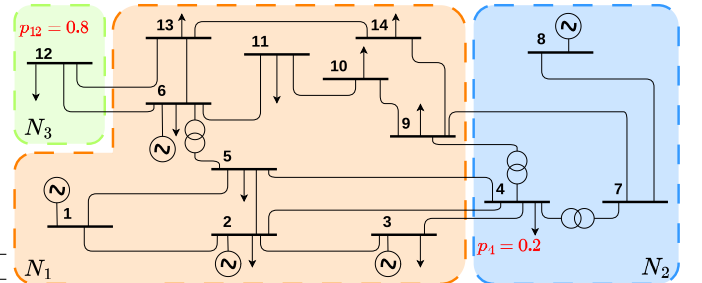


Fig. 4. 14-bus network and the optimal islands found by the exact and MCTS algorithms when buses 4 and 12 are attacked with $p_4 = 0.2$ and $p_{12} = 0.8$.

TABLE III
ATTACK ON BUSES 4 AND 12 WITH $p_4 = 0.2$ AND $p_{12} = 0.8$.

Methods	Utility (MW)				Time (s)	
	Mean	SD	Max	Min	Mean	SD
MCTS-50	200.02	0.84	202.32	199.52	0.85	0.08
MCTS-500	202.87	3.73	208.93	199.52	2.58	0.24
MCTS-5000	206.08	3.72	208.93	200.12	7.50	0.87
Exact	208.93	—	—	—	20.30	0.12

the number of samples, approaching the optimal value found by the exact algorithm. Both the maximum and minimum utilities obtained also increase with larger sample sizes. With 5000 samples, MCTS finds the optimal solution in at least one of the 20 runs. Its computation time increases with the number of samples, yet even with 5000 samples, it remains significantly lower than that of the exact algorithm.

Table III presents the performance for a multi-node attack scenario, where buses 4 and 12 are attacked with $p_4 = 0.2$ and $p_{12} = 0.8$. As in the previous case, the mean utility of MCTS increases with the number of samples. In this scenario, the optimal solution was found by some MCTS runs with 500 samples. The computation time also exhibits a trend similar to the previous case, increasing with the number of samples while remaining significantly lower than that of the exact algorithm for up to 5000 samples. Figure 4 shows the 14-bus network and the optimal solution found by the exact algorithm and also by MCTS with 500 and 5000 samples. The solution forms three islands, where bus 12, with a high attack probability, is isolated, and bus 4, with a low probability, is joined by buses 7 and 8, while the rest of the network is completely separated from the risk of failure due to a cyber attack.

A drawback of the stochasticity in MCTS is that some runs may yield relatively poor solutions, as reflected in the variability of observed utilities. However, the results also demonstrate that MCTS can find (near-)optimal solutions with substantially lower computation times. This suggests that multiple MCTS instances could be executed in parallel and the best solution selected, providing a practical means of improving solution quality while retaining the computational advantages of the approach.

VI. CONCLUSION

In this work, we proposed preventive controlled islanding as a resilience-enhancing response strategy against cyber security threats when the emerging attack locations and their associated probabilities are known. The problem was modeled as graph-constrained coalition formation and solved using exact and adapted MCTS algorithms. Empirical results on the IEEE 14-bus system showed that optimal solutions account for both attack locations and their probabilities and are not constrained by a predetermined number of islands. Furthermore, the MCTS algorithm found (near-)optimal solutions while requiring substantially less computation time than the exact algorithm, and running MCTS instances in parallel increases the likelihood of finding the optimal solution. Future work will consider preventive islanding on larger networks, for

which the exact algorithm becomes computationally infeasible. Although the MCTS algorithm may still find high-quality solutions, the required number of samples is expected to increase. Consequently, the development of more efficient algorithms remains an important direction for future research. Additionally, the proposed formulation could be extended to incorporate alternative operational constraints and to other power systems, such as distribution networks.

REFERENCES

- [1] M. J. Assante, R. M. Lee, and T. Conway, "Analysis of the cyber attack on the Ukrainian power grid," E-ISAC, Tech. Rep., 2016.
- [2] —, "Modular ICS malware," E-ISAC, Tech. Rep., 2017.
- [3] R. Dasgupta, M. Pramanik, P. Mitra, and D. R. Chowdhury, "Intrusion detection for power grid: a review," *Int. J. Inf. Secur.*, vol. 23, no. 2, pp. 1317–1329, 2024.
- [4] A. Presekal, A. Štefanov, V. S. Rajkumar, and P. Palensky, "Attack graph model for cyber-physical power systems using hybrid deep learning," *IEEE Trans. Smart Grid*, vol. 14, no. 5, pp. 4007–4020, 2023.
- [5] M. Massaoudi, M. Ez Eddin, A. Ghrayeb, H. Abu-Rub, and S. S. Refaat, "Advancing coherent power grid partitioning: A review embracing machine and deep learning," *IEEE Open Access J. Power Energy*, vol. 12, pp. 59–75, 2025.
- [6] T. E. K. Zidane, Z. A. Muis, W. S. Ho, Y. Zahraoui, A. S. Aziz, C.-L. Su, S. Mekhilef, and P. Elia Campana, "Power systems and microgrids resilience enhancement strategies: A review," *Renew. Sustain. Energy Rev.*, vol. 207, p. 114953, 2025.
- [7] A. Štefanov, N. Cibi, M. Hoeve, S. Talwar, J. Godefrooi, and D. van Hofwegen, "Cyber security of Control Room of the Future using artificial intelligence to strengthen power grid cyber resilience to advanced persistent threats," To appear in *IEEE Power Energy Mag.*, Oct. 2026.
- [8] F. Bistaffa, A. Farinelli, J. Cerquides, J. Rodríguez-Aguilar, and S. D. Ramchurn, "Algorithms for graph-constrained coalition formation in the real world," *ACM Trans. Intell. Syst. Technol.*, vol. 8, no. 4, 2017.
- [9] R. D. Christie, "Power systems test case archive," 1993, <https://labs.ece.uw.edu/pstca/>.
- [10] S. Paul, Z. Ni, and C. Mu, "A learning-based solution for an adversarial repeated game in cyber-physical power systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 11, pp. 4512–4523, 2020.
- [11] L. Niu, D. Sahabandu, A. Clark, and R. Poovendran, "A game-theoretic framework for controlled islanding in the presence of adversaries," in *Decision and Game Theory for Security*. Springer, 2021, pp. 231–250.
- [12] S. Basumallik, S. Eftekharnajad, and M. Fardad, "Controlled islanding under complete and partial false data injection attack uncertainties against phasor measurement units," *Energies*, vol. 15, no. 15, 2022.
- [13] S. Hashemi, M. Panteli, V. S. Rajkumar, and A. Štefanov, "Operational mitigation of cyberattack-induced cascading failures in power systems," in *Proc. Int. Conf. Smart Energy Syst. Technol. (SEST)*, 2024, pp. 1–6.
- [14] D. Csersik and L. Á. Kóczy, "Efficiency and stability in electrical power transmission networks: A partition function form approach," *Netw. Spat. Econ.*, vol. 17, no. 4, pp. 1161–1184, 2017.
- [15] N. Mohseni, T. Morstyn, C. O'Meara, D. Bucher, J. Nüßlein, and G. Cortiana, "Evidence of quantum scaling advantage in approximate optimization for energy coalition formation with 100+ agents," *Quantum Sci. Technol.*, vol. 11, no. 1, 2025.
- [16] J. Nüßlein, D. Schuman, D. Bucher, N. Mohseni, K. Ghosh, C. O'Meara, G. Cortiana, and C. Linnhoff-Popien, "Towards less greedy quantum coalition structure generation in induced subgraph games," in *Proc. IEEE Int. Conf. Quantum Comput. Eng. (QCE)*, vol. 02, 2024, pp. 28–33.
- [17] X. Kong and X. Tong, "Monte-Carlo tree search for graph coalition structure generation," in *Proc. Int. Congr. Image Signal Process., BioMed. Eng. Informatics (CISP-BMEI)*, 2020, pp. 1058–1063.
- [18] C. Coffrin, R. Bent, K. Sundar, Y. Ng, and M. Lubin, "PowerModels.JL: An open-source framework for exploring power flow formulations," in *Proc. Power Syst. Comput. Conf. (PSCC)*, 2018, pp. 1–8.
- [19] J. Carpentier, "Contribution to the economic dispatch problem," *Bull. Soc. Fr. Electr.*, vol. 3, no. 8, pp. 431–447, 1962.
- [20] Z. Wang, A. Younesi, M. V. Liu, G. C. Guo, and C. L. Anderson, "AC optimal power flow in power systems with renewable energy integration: A review of formulations and case studies," *IEEE Access*, vol. 11, pp. 102 681–102 712, 2023.