A Binary Decision Diagram Based Cascade Prognostics Scheme For Power Systems

Ajay Chhokra, Saqib Hasan, Abhishek Dubey and Gabor Karsai
Vanderbilt University, Nashville, TN 37212, USA
Email: {ajay.d.chhokra; saqib.hasan; abhishek.dubey; gabor.karsai}@vanderbilt.edu

Published in the proceedings of the American Control Conference 2020 (ACC 2020)

***Abstract*— Cascading outages in power systems is a rare, but important phenomenon with huge social and economic implications. Due to the inherent complexity and heterogeneity of components in power system, analysis and prediction of the current and future states of the system is a challenging task. In this paper, we address prognosis of cascading outages in power systems by employing a novel approach based on reduced ordered binary decision diagrams. We present a systemic way of synthesizing these decision diagrams based on a simple cascade model. We also describe a workflow for finding the emergency load curtailment actions as a part of the mitigation strategy. In the end, we show the applicability of our approach using the standard IEEE 14 bus system.**

***Index Terms*— Binary Decision Diagrams, Contingency Analysis, Nonlinear Optimization, Sensitivity Analysis, Load Curtailment**

I. INTRODUCTION

Power system equipment is constantly exposed to dynamic environments caused due to changing loading conditions, physical degrading of the components and external faults such as earthing and winding faults. The safety of the system is ensured by a large infrastructure of *protection system assemblies*. A protection system assembly is composed of instrument transformers, intelligent software enabled protection relays and high-voltage circuit breakers. Relays sample the scaled down voltage and current signals from instrument transformers and based on embedded relay logic ascertain the presence of a fault. On detecting the presence of faulty conditions, the relay sends a tripping signal to the breaker which isolates the faulty component from the system. However, due to a lack of system wide perspective and hidden faults (incorrect settings), the actions of protection devices have been known to cause cascading outages [1]. A cascading outage is defined as an uncontrolled loss of any system facilities or load as a result of fault isolation. Such cascading outages in power grids successively weaken the system by increasing stress on other components and can lead to complete blackouts.

There are two stages associated with cascading outages. The first stage is the steady state progress stage: a period of slowly evolving successive events. The timescale associated with the evolution of events in this period is of the order of min-hrs. The second stage is the transient progress stage: a fast transient process marked by uncontrolled tripping of a large number of generators, transmission lines and transformers. The timescale associated with this period is of the order seconds to minutes. During the first stage, system operators evaluate system conditions against different grid stability criteria to identify state trajectories and take some control actions to improve the operating conditions and prevent the possible cascading outages. The control cost is minimal compared with the massive cost of cascading outages. The current industry practice involves performing on demand analysis of cascades using different simulation models. However, these simulations take a considerable amount of time to finish and data generated by these complex simulation models is difficult to analyze in a timely manner. This increases the line operators' response time to anticipate the future state of the system. Thus, an efficient surrogate model is required that can quickly classify the stability of the system and provides cascade progression in case the system is deemed unstable.

In this paper, we propose a novel binary decision diagram based *prognostics* methodology. By prognostics, we imply predicting the future state of the system in terms of outages. We model cascade progression as a transition relation and use reduced order binary decision diagrams (BDDs) to encode these progressions for memory efficient and faster look-up. We also present optimal load curtailment as cascade mitigation measures where load curtailment actions are identified at run time. The novelty of our approach lies in the data-driven aspect of both cascade prognostics and mitigation. The rest of the paper is organized as follows: Section II illustrates the state of the art related to cascade prognostics and mitigation. Section III describes our approach by discussing identification of critical component outages, prognostics and mitigation followed by results and conclusions in sections IV and V respectively.

II. RELATED RESEARCH

A. Prognostics Methodologies

Current industry practice is to determine (offline) critical components of the power system such that their outages can successively weaken the system leading to blackouts. The process of finding these critical component outages is called *contingency analysis*. N-1 contingency analysis refers to a single component outage out of total N system components. The cascades in the past have been caused due to the interaction of more than 1 independent component outage.

Hence, it is required to perform high-order (N-k) contingency analysis where k is the number of initial outages in a system with N components. It is well understood that calculating higher order contingencies are infeasible as the total number of combinations grows exponentially.

A number of methodologies exist in the literature that tries to identify contingencies in a power network. We categorize these techniques into the following two categories:

1) Topology Based: There is a substantial amount of research being done into understanding cascading phenomenon using *topological contagion* models [2]. These threshold based contagion models have been deemed useful in comprehending problems like disease spreading [3] and social influence spreading [4]. Similar approaches have been proposed for finding contingencies in power systems [5]. Contagion models are based on the assumption that a component outage affects only nearby components. However, the premise of cascade progression being a local phenomenon does not hold well in power systems.

2) Simulation Based: The second approach uses simulation models of power grids to understand the cascade propagation and identify critical component outages that can severely impact the power system. This approach is used by line operators in offline and online settings. Examples of such simulation models are DCSIMSEP [6], Oak Ridge-PSERC-Alaska (OPA) [7], Manchester model [8], TRELSS [9] and COSMIC [10]. However, these simulations take a considerable amount of time to finish. Moreover, the generated data from these simulations is complicated and thus difficult to understand and summarize quickly.

B. Mitigation Strategies

Once the cascade conditions have been identified, predefined actions such as load shedding can be used to suppress the cascading effects of overloads, voltage and frequency instabilities. An alternative approach is to curtail a percentage of the load instead. Curtailment provides an effective means of handling the cascade effects without disconnecting the complete load. For example, the cascading failures during the blackout of Aug 2003 in the USA could have been avoided by removing a relatively small amount of load in the Cleveland area [11], [12]. However, the most effective load curtailment is not always obvious. Line operators rely on optimal power flow algorithm to identify the suitable generator or load re-dispatch actions. General practice is to use simple linear programming to find minimalistic load shedding actions that can prevent the progression of a cascade. But the linear approximation of the underlying system can be misleading and can result in incorrect load management. A number of approaches based on model predictive control [11], [13], [14] have been proposed that tackles problems of voltage collapse and successive branch outages due to overloads. However, above mentioned model predictive control strategies are not always guaranteed to provide an optimal solution because of the limitation of the underlying approximation of the mathematical model and limited number of control actions as per the control horizon.

The work presented in this paper is the extension of our previous work [15] that proposes a systematic approach of finding load curtailment actions in an offline setting. However, this methodology covers 1) identifying critical state of the system 2) encoding the blackout causing states as a transition relation using binary decision diagrams and 3) calculating mitigating actions at run-time.

III. Approach

Our approach utilizes reduced order binary decision diagrams [16] (BDDs) to encode different blackout causing outages (contingencies). The advantage of using BDDs is their ability to encode complex behaviors that can be used in reasoning about cascade progression efficiently while incurring small memory footprint and at the same time allowing fast access time. The proposed cascade prognosis methodology consists of two phases 1) *Offline* and 2) *Online*. Initially, in the offline phase, critical outages or contingencies are identified followed by storing these contingencies and their respective progression in BDDs. Whereas in online phase, actual prognosis is done using BDDs created in the previous stage and load curtailment actions are calculated based on the current state of the system.

Cascading outages in power system are primarily caused by production and demand imbalance. The initiating event can be generator (source), transmission line or transformer (branch) outage caused by fault isolation or planned maintenance. The initial outages can increase stress in the rest of the system causing secondary effects in terms of branch overloads, bus voltage fluctuations and frequency instability. These secondary effects can lead to more outages by the action of protection devices which can further destabilize the system leading to blackouts. If the secondary effects can be removed by curtailing a part of the load, then cascading outages can be prevented. We formulate the load curtailment as a non-linear optimization problem and utilize OpenMDAO [17] that uses external steady state simulator, OpenDSS [18] to find optimal control actions. The optimization framework, OpenMDAO acts as an orchestrator for finding voltage and current gradients by triggering OpenDSS to solve the power flow equations at different values of load demands and generator power injections. Our approach is different from the existing approaches as it does not assume linear power flow model. The online and offline phases are discussed in more detail in the following subsections.

A. Identification of Contingencies

We developed a simple cascade simulation model (based on steady state calculations) that successively solves the power flow (using OpenDSS) by removing the overloaded branches from the system after the initial component outages. The simulation keeps on tripping the overloaded branches till a blackout situation is reached or there are no more secondary effects (overloads) in the system. This cascade simulation model caters to slowly progressing cascades that eventually lead to blackouts involving overloads. We have adopted a conservative approach where all the secondary effects of initial outages are mitigated through the existing (pre-defined) protection schemes that isolate the overloaded components from the system.

Algorithm 1 Algorithm for finding critical N-k contingencies

```
Input: Model, k, Branch
Output: T, TR
A ← choose(Branch, k)                          ▷ Generating contingency list
j ← 1
for j ≤ (|Branch| choose k) do
    Prev ← A[j], Next ← ∅, Temp ← ∅, Start ← A[j]
    Model.apply_contingency(Prev)              ▷ Applying j^th contingency
    while True do
        if Model.check_blackout() then
            T ← T ∪ Start                      ▷ Save contingency
            TR ← TR ∪ Temp                     ▷ Save the sequence of branch outages
            break
        else
            Next ← Model.get_overloads()       ▷ Identify overloaded branches
            if Next ≠ ∅ then
                Temp ← Temp ∪ (Next ∪ Prev, Prev)
                Prev ← Temp
                Model.trip_branches(Next)      ▷ Tripping overloaded branches
            else
                break
            end if
        end if
    end while
    j ← j + 1                                  ▷ Iterate to next contingency
end for
```

Listing 1 shows the underlying algorithm to find N-k contingencies. The input parameter of the algorithm includes a OpenDSS model (*Model*), an integer representing the order of contingencies (*k*) and a set of all branch labels (*Branch*). The output of the algorithm consists of two sets T, TR that represents a collection of initiating events and their respective progressions. The set, $T = \{s_1, s_2, ..., s_n\}$ is a collection of all contingencies that can cause blackout, where s_i is some combination of branch outages. The set, $TR_{s_1} = \{(s_1, s_2), (s_2, s_3), ...(s_i, s_j)\}$ represents the progression of cascade caused due to s_1, where s_i represents the initial branch outages and s_j implies the branch outages as a consequence s_i. The algorithm starts with tripping k lines at random and solving the power flow to update the branch currents and bus voltages. The second step is to check for the blackout criteria. The blackout criteria is configurable in terms of the percentage of the original load (demand) that is not operational. For a blackout criteria of 40%, if more than 40% of the net system load demand cannot be satisfied in a given state, then the system is considered to have reached blackout. If the system is not in a blackout state, then secondary effects of the branch outages are investigated by checking the overloads in rest of the system. If no overloads are found then, the system is considered to have reached a safe state from where it cannot reach blackout. On the other hand, if some secondary overloads are present, the transition relation, represented by $Temp$ is updated followed by tripping all those branches. After branch tripping, the blackout criteria is checked again and the process repeats until a blackout state is reached or the system reaches a stable state (no overloads).

B. *Efficient Storage Mechanisms*

We employ compact and efficient data structure, ordered Binary Decision Diagrams (BDDs) to store the progressions of cascading outages. A binary decision diagram is a data structure that is used to represent boolean functions. A BDD is a directed acyclic graph that consists of two types of nodes, A) *Decision Nodes*: Each decision node represents a boolean variable, V_i, and has two child nodes, *high* and *low*. The edge from node, V_i to a low (or high) child represents an assignment of V_i to 0 (1). B) *Terminal Nodes*: There are two types of terminal nodes called 0-terminal and 1-terminal. A path from the root node to the 1-terminal (0-terminal) represents a variable assignment for which the represented Boolean function is true (false). A reduced ordered BDD has a fixed ordering i.e different variables exists in the same order along different paths and has an important *canonicity* property i.e. for a fixed variable ordering, each boolean function has a unique representation. On an abstract level, these BDDs are used as a compressed representation of sets and transition relations that has relatively small memory footprint and allow fast retrieval of the encoded information as operations are performed directly on the compressed form. The progression of a cascade is represented by state transitions, where state defined by the component outages. Based on the initiating outages identified using algorithm 1 two BDDs are created with the following functionalities :

- First BDD labeled as, B_T, stores the set of initiating events that will cause cascading outages in the rest of the system leading to a complete system blackout.
- The Second BDD, labeled as, B_{T_R} stores the progression of all the initiating events captured by B_T as a translation relation.

1) BDD encoding of collection of initiating events: Let T be the set of branch outages identified by the offline N-k contingency analysis. Each element in T represents a collection of initiating events i.e. independent branch outages that can trigger a cascading phenomenon leading to a blackout. Since the main objective is to encode these sets of line outages, every element of T can be represented by a unique boolean vector $(v_1, v_2, v_3, ..., v_n)$, each $v_i \in 0, 1$, of length equal to the number of branches in the system. Then $T \subseteq S$ can be represented by a characteristic function $f_T : \{0, 1\}^n \to \{0, 1\}$ which maps a particular evaluation of $(v_1, v_2, v_3, ..., v_n)$ to either 0 or 1, where S is the power set. For each $s \in S$, if the value mapped by f_T is 1 then $s \in T$ otherwise s in not the member of T.

We define a labeling function for S, $L(S) : S \to P(Branch)$, where $Branch$ is a set of branch outages, say $(tl_1, tl_2, tl_3, ..., tl_n)$ associated to an initiating event combination, $s \in S$. Hence s can be represented by a boolean vector $(v_1, v_2, v_3, ..., v_n)$ where v_i is 1 if $tl_i \notin L(s)$. Here $v_i = 1$ means the power is flowing through branch tl_i i.e. all the breakers associated to the branch are closed whereas the value 0 implies no power flow. As a BDD, the initiating event combination, $s \in S$ is represented by the boolean function, $l_1 \cdot l_2 \cdot l_3 ... \cdot l_n$ where l_i is tl_i if $tl_i \notin L(s)$ otherwise $\overline{tl_i}$. The set T can be represented by the boolean function f_T, $(l_{11} \cdot l_{12} \cdot l_{13} ... \cdot l_{1n}) + (l_{21} \cdot l_{22} \cdot l_{23} ... \cdot l_{2n}) + ... + (l_{j1} \cdot l_{j2} \cdot l_{j3} ... \cdot l_{jn})$ where $(l_{k1} \cdot l_{k2} \cdot l_{k3} ... \cdot l_{kn})$ represents the initiating event set s_k.

2) BDD encoding of progression of cascading outage: The progression of cascade can be modeled by transition relation. A translation relation is a boolean function, $f_{T_R}: S \times S \rightarrow \{0,1\}$, which outputs 1 if there exists a transition between two given states otherwise 0. Similar to T, set of valid transitions, T_R can be viewed as subset of all possible transitions, i.e. $T_R \subseteq S \times S$. An element $t \in T_R$ implies a transition from state s to s' and is represented by a pair of boolean vectors $((v_1, v_2, v_3, ..., v_n), (v'_1, v'_2, v'_3, ..., v'_n))$ where v_i is 1 if $tl_i \notin L(s)$ and 0 otherwise; and similarly, v'_i is 1 if $tl_i \notin L(s')$. A single transition link can be represented by a boolean function $(l_1 \cdot l_2 \cdot l_3 ... \cdot l_n) \cdot (l'_1 \cdot l'_2 \cdot l'_3 ... \cdot l'_n)$ and the complete set T_R can be represented as disjunction of such formulas as shown in the case of initiating events.

C. Identifying Cascade Progression

After creating these BDDs, the next task is to evaluate the current system state s_t, represented by a boolean vector $(v_1, v_2, ..., v_n)$, where n is the number of branches, is a member of the set T. If $s_t \in T$, then the progression of s_t can be calculated by finding the set of reachable states from s_t under a given transition relation, f_{T_R}. The operation of finding the set of states reachable from a given state is called image computation and the process of calculating image iteratively till a fixed point is reached is a fundamental step in many state exploration algorithms. The algorithm 2 shows the algorithm in determining the cascade progression for a given state of the system.

Algorithm 2 Algorithm for determining the evolution of current state, S_0

```
Input: S_0 = (v_1, v_2, ..., v_n) ; B_T ; B_{T_R}
Output: S_reach
Initialize: S_reach = φ
if Evaluate(B_T, S_0) == True then
    i = 0
    while S_i ≠ φ do
        S_reach = S_reach ∪ S_i                      ▷ Update reachable set
        S_{i+1} = Image(B_{T_R}, S_i)\S_reach         ▷ Identify new reachable states
        i = i+1
    end while
end if
```

The algorithm 2, requires a set of line outages (S_0), the BDDs B_T, B_{T_R}. The output of the algorithm is a sequence of states reachable from S_0 if $S_0 \in T$. The Evaluate(B_T, S_0) function checks whether S_0 is a member of set T represented by B_T. IF yes, then recursively next states are found and added to the set S_{reach} until fixed point is reached when for a given state S_{i-1} no new next state is defined i.e S_i is empty.

D. Cascade Mitigation

Cascading outages can be mitigated by adjusting the load demand of the system such that secondary branch overloads disappear. However, the amount of load curtailment should be minimized as a large difference between the power supplied by the generators and load demand can increase instability and leading to system collapse. We formulate identification of load curtailment as an optimization problem, described in equations (1)-(6), where, $\mathbf{L}$ (ohms) is a vector of load demands of size M, $\Delta\mathbf{L}$ is a vector of decision variables, such that ΔL_i denotes the (ratio) curtailment of load $L_i \in \mathbf{L}$ by $\Delta L_i \cdot |L_i|$ ohms. $\mathbf{I}$ is the collection of branch currents in the system.

$$\min_{\Delta\mathbf{L}} \quad \sum_{i=0}^{M} w_i \cdot \Delta L_i \cdot |L_i| \tag{1}$$

$$0 \leq |I_j| \leq I_j^{Max}, \quad \forall I_j \in \mathbf{I} \tag{2}$$

$$0 \leq \Delta L_i \leq 1, \quad \forall L_i \in \mathbf{L} \tag{3}$$

$$\Phi_{L_i} = \Phi_{(1-\Delta L_i)\cdot L_i}, \quad \forall L_i \in \mathbf{L} \tag{4}$$

$$\sum_i (\Delta L_i) \cdot |L_i| \leq L_{max}^{total}, \quad \forall L_i \in \mathbf{L} \tag{5}$$

$$\mathbf{I} = f(\Delta\mathbf{L}) \tag{6}$$

The objective function is the weighted sum of all load curtailments as shown in equation 1 where weights, w_i models the importance associated with a load. For instance, critical loads can be establishments of national or societal importance such as hospitals and government buildings having large w_i. The inequality constraint described in equation 2 ensures no branch overloads are present in the final solution, where I_j^{Max} defines an upper limit on the current that can flow through a branch. The inequality constraints 3 describes the extent to which individual loads can be changed. The equality constraint 4 ensures the power factor, Φ of all loads are maintained i.e real and reactive load is shed in equal proportions. The inequality constraint 5 describes the upper bound on the total load, L_{max}^{total} that can be shed from the system. In the current implementation, it is 20% of the total system load. The function, f in equation 6 models the load flow equations. It updates the branch currents ($\mathbf{I}$) according to the change in system loads.

The convergence of optimization algorithm depends upon the size of the input search space i.e. number of design variables and their initial estimates. In order to speed up the convergence our proposed optimization framework performs sensitivity analysis to find filter out the loads that do not affect a given branch overload and obtain initial estimates. The sensitivity analysis can be broken into 3 sequential steps described as follows:

1) Data point generation: In this step, the effect of a varying absolute value of load demand, L_i on all the branch currents, $\mathbf{I}$ is observed. The load (L_i) vs branch current (I_j) data points are stored for learning a regression model. We have used Full Factorial Design of Experiment (DoE) analysis that uniformly samples the input space i.e. range [0, L_i] for each load $L_i \in \mathbf{L}$. In our implementation, 100 data points are considered for each load.

2) Regression Analysis: This step involves finding equation parameters (slope and intercept) for branch current vs load change data points generated in the previous step. It is safe to assume linear relationship between branch currents and load demand since the power factor remains constant. The sensitive loads can be classified by observing the slope of the equation, $|I_j| = m|L_i| + C$, where $|I_j|$ are the absolute value of current flowing through j^{th} branch and $|L_i|$ is the i^{th} load (magnitude) ; and m, c are the equation parameters. For

TABLE I: Timing analysis for IEEE 14 Bus system

Parameters	IEEE 14 Bus System
Variable count	20
B_T, B_{T_R} *construction time (secs)*	18.80
Average time for true positive cases (secs)	0.006
Average time for true negative cases (secs)	3.5×10^{-5}

a given load, if the slope is positive for any branch current, then the load is considered to be sensitive.

3) Starting Point Estimation: In this step, we estimate the starting value for each decision variable, $\Delta L_i \in \Delta\mathbf{L}$ as described in the equations (7) and (8), where I_j is the current in j^{th} branch, (C_j^i, m_j^i) are parameters of the learned regression model that relates branch current I_j and load L_i. S_i is the set of branches (indices) for which the load L_i is classified as sensitive.

$$\mathbf{L}_i^{'} = [\frac{I_j - C_j^i}{m_j^i}], \quad \forall j \in S_i \tag{7}$$

$$\Delta L_i = \frac{|L_i| - \min(\mathbf{L}_i^{'})}{|L_i|} \tag{8}$$

IV. Results

In order to validate the accuracy of our approach, IEEE 14 bus system [19] is used. The system consists of 14 buses, 5 generators, 11 loads and 20 branches (transmission lines and transformers). As per the algorithm 1, a total of 400 critical and 600 non-critical outages are identified with k ranging from [1,3]. The former set of 400 outage combinations are referred to as true positive cases and the latter are called true negatives. The blackout criteria used is 40% of total system load.

Table I summaries the results of the experiments. The size of the boolean vector (state) is 20 (equal to number of branches). As shown in the Table I, a small amount of overhead, 18 secs is added for constructing BDDs. On an average, fixed point computation i.e. identification of cascade progression, for true positive cases, takes 7 milliseconds whereas 0.03 milliseconds for true negatives. Figure 1(Left) show the response time of all the true positive and negative cases. These experiments are performed on a 1.7 GHz Intel Core i7 machine with 8 GB RAM.

The optimization routine was able to find a solution for all 400 cases with an average of 29 iterations. Figure 3 lists the % load demand reduction. In all cases, load curtailment is restricted to less than 20 % of the net system load (constraint 6) with an average load reduction of approximately 8%. Table II lists the solver parameters used for the experiments.

TABLE II: Solver parameters

Parameter	Value
Problem type	Non Linear
Solver	Sequential Least SQuares Programming
Derivative Calculation	Forward finite difference
Step Size	2500.00
Max Iterations	1000

Scalability Analysis: Since power systems are large networks its imperative to discuss the impact of scale on

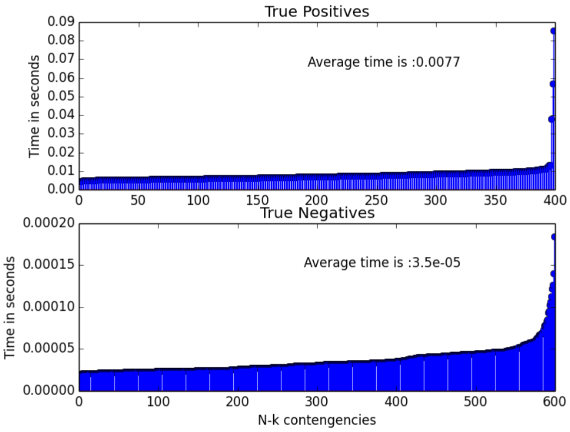

Fig. 1: Response Time for set membership and fixed point calculation i.e. cascade progression for 1000 different combinations of branch outages for IEEE 14 Bus System. The figure on top shows the time taken for 400 actual cascade causing outages. The figure on the bottom shows the time taken to respond to the 600 True Negative or safe outage combinations

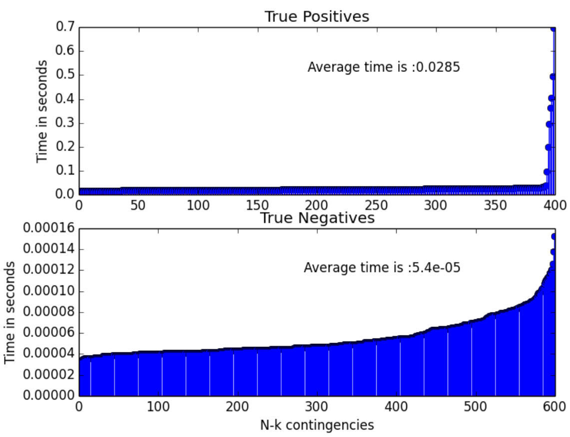

Fig. 2: Response Time for set membership and fixed point calculation for IEEE 39 Bus System

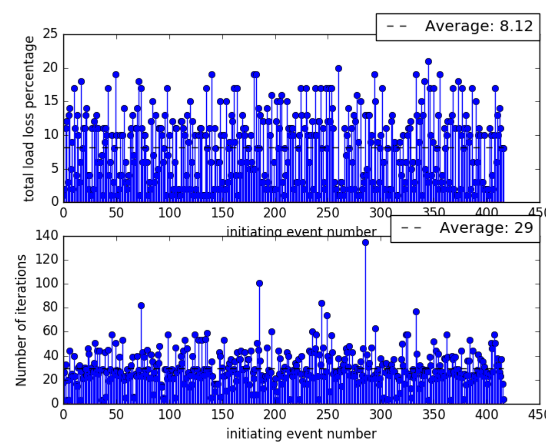

Fig. 3: The figure on the top shows the net load loss (percentage) due to load control actions and the bottom one shows the number of iterations taken by the optimization engine to find a solution

our approach. The presented approach consists of 3 major computational tasks 1) *N-k Contingency Analysis*, for small values of k (1, 2, 3 in our case), this process has approximately polynomial run time complexity as the number of combinations increases polynomially with increase in the size of network (N) as well as the time required for solving power flow [20]. 2) *BDD encoding and prognostics*, BDD encoding of singe outage depends upon the size of the network. In order to find the relationship between number of branches and set-membership time (identifying whether a given state is critical or not), 400 true positive and 600 true negative outages are identified for a larger IEEE 39 bus system [21] with 46 branch variables. As the number of branch variables doubles, the set-membership time roughly doubled as shown in figure 2. 3) *Optimization*, the number of constraints (one per branch) and design or control variables (loads) increases linearly with the increase in the size of the power network. It is well known that the performance of the optimizer is greatly affected by starting point estimate and the size of the input space of the problem. The sensitivity analysis routine uses full factorial based analysis to estimate a starting point for each load to prevent cascade. For large systems the number of sensitive loads might be very large. A bound on number of control variables (loads) can be placed that can reduce the search space for the optimization problem. If the number of sensitive loads is reduced to 2 (Average number of control variables in first experiment was 7), then the average number of iteration have reduced from 29 to 25 with a slight increase in percentage load reduction from 8.12 to 8.19 %.

V. Conclusions

In this paper, we discussed the problem of simulating fault cascades in power systems and presented a novel way of storing the useful information from cascade simulations. We showed with the help of simple cascade model, the generation of binary decision diagrams. We presented a detailed algorithm to encode the results of N-k contingencies as BDDs and utilization of BDDs for prognosis of future cascades (if any) given the current system state. We also presented an extensible optimization methodology based on OpenMDAO and OpenDSS to identify load control actions to avoid cascading outages. In the end, we presented the timing overheads in the construction of the two BDDs followed by the scalability analysis. As part of the future work, we would create a database of BDDs for different probable loading profiles by considering the load distributions and past data.

Acknowledgments

This work is funded in part by the National Science Foundation under the award numbers 1329803, 1840052 and 1818901. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of NSF. The authors will like to thank Rishabh Jain and Srdjn Lukic for their help and discussions related to the work presented here.

References

[1] R. Baldick, B. Chowdhury, I. Dobson, Z. Dong, B. Gou, D. Hawkins, H. Huang, M. Joung, D. Kirschen, F. Li, *et al.*, “Initial review of methods for cascading failure analysis in electric power transmission systems.” IEEE, 2008, pp. 1–8.

[2] D. J. Watts, “A simple model of global cascades on random networks,” *Proceedings of the National Academy of Sciences*, vol. 99, no. 9, pp. 5766–5771, 2002.

[3] L. Hébert-Dufresne, P.-A. Noël, V. Marceau, A. Allard, and L. J. Dubé, “Propagation dynamics on networks featuring complex topologies,” *Physical Review E*, vol. 82, no. 3, p. 036115, 2010.

[4] P. S. Bearman, J. Moody, and K. Stovel, “Chains of affection: The structure of adolescent romantic and sexual networks1,” *American journal of sociology*, vol. 110, no. 1, pp. 44–91, 2004.

[5] P. Crucitti, V. Latora, and M. Marchiori, “A topological analysis of the italian electric power grid,” *Physica A: Statistical mechanics and its applications*, vol. 338, no. 1, pp. 92–97, 2004.

[6] M. J. Eppstein and P. D. Hines, “A random chemistry algorithm for identifying collections of multiple contingencies that initiate cascading failure,” *IEEE Transactions on Power Systems*, vol. 27, no. 3, pp. 1698–1705, 2012.

[7] B. A. Carreras, V. E. Lynch, I. Dobson, and D. E. Newman, “Critical points and transitions in an electric power transmission model for cascading failure blackouts,” *Chaos: An interdisciplinary journal of nonlinear science*, vol. 12, no. 4, pp. 985–994, 2002.

[8] D. P. Nedic, I. Dobson, D. S. Kirschen, B. A. Carreras, and V. E. Lynch, “Criticality in a cascading failure blackout model,” *International Journal of Electrical Power & Energy Systems*, vol. 28, no. 9, pp. 627–633, 2006.

[9] M. Bhavaraju and N. Nour, “Trelss: A computer program for transmission reliability evaluation of large-scale systems,” Electric Power Research Inst., Palo Alto, CA (United States); Public Service Electric and Gas Co., Newark, NJ (United States), Tech. Rep., 1992.

[10] J. Song, E. Cotilla-Sanchez, G. Ghanavati, and P. D. Hines, “Dynamic modeling of cascading failure in power systems,” *IEEE Transactions on Power Systems*, vol. 31, no. 3, pp. 2085–2095, 2016.

[11] P. Hines and S. Talukdar, “Controlling cascading failures with cooperative autonomous agents,” *International journal of critical infrastructures*, vol. 3, no. 1-2, pp. 192–220, 2006.

[12] A. Berizzi, “The italian 2003 blackout,” in *Power Engineering Society General Meeting, 2004. IEEE*. IEEE, 2004, pp. 1673–1679.

[13] J. Jung, C.-C. Liu, S. L. Tanimoto, and V. Vittal, “Adaptation in load shedding under vulnerable operating conditions,” *IEEE Transactions on Power Systems*, vol. 17, no. 4, pp. 1199–1205, 2002.

[14] M. R. Almassalkhi and I. A. Hiskens, “Model-predictive cascade mitigation in electric power systems with storage and renewablespart i: theory and implementation,” *IEEE Transactions on Power Systems*, vol. 30, no. 1, pp. 67–77, 2015.

[15] A. Chhokra, A. Kulkarni, S. Hasan, A. Dubey, N. Mahadevan, and G. Karsai, “A systematic approach of identifying optimal load control actions for arresting cascading failures in power systems,” in *Proceedings of the 2nd Workshop on Cyber-Physical Security and Resilience in Smart Grids*, 2017.

[16] S. Akers, “Binary decision diagrams,” *IEEE Transactions on Computers*, vol. 27, no. 06, pp. 509–516, jun 1978.

[17] J. Gray, K. T. Moore, and B. A. Naylor, “Openmdao: An open source framework for multidisciplinary analysis and optimization,” in *AIAA/ISSMO Multidisciplinary Analysis Optimization Conference Proceedings*, vol. 5, 2010.

[18] R. C. Dugan, “Reference guide: The open distribution system simulator (opendss),” *Electric Power Research Institute, Inc*, 2012.

[19] [Online]. Available: http://www2.ee.washington.edu/research/pstca/pf14/pg_tca14bus.htm

[20] F. L. Alvarado, “Computational complexity in power systems,” *IEEE Transactions on Power Apparatus and Systems*, vol. 95, no. 4, pp. 1028–1037, July 1976.

[21] T. Athay, R. Podmore, and S. Virmani, “A practical method for the direct analysis of transient stability,” *IEEE Transactions on Power Apparatus and Systems*, no. 2, pp. 573–584, 1979.