Improving system resilience through formal verification of transactive energy controls

Alan Ransil
MIT Energy Initiative
Massachusetts Institute of Technology
Cambridge, MA
Protocol Labs Research
Protocol Labs
San Francisco, CA
ransil@mit.edu

Michael Hammersley
Protocol Labs Research
Protocol Labs
San Francisco, CA
michael.hammersley@protocol.ai

Francis M. O'Sullivan
MIT Energy Initiative
Massachusetts Institute of Technology
Cambridge, MA
frankie@mit.edu

Abstract—**Formal verification tools such as TLA+ allow errors to be uncovered through exhaustive exploration of reachable states, and are the gold standard for ensuring resilience in software systems. In particular, these methods can be used to identify error states emerging from precise interactions between multiple subsystems that would occur only after long periods of testing, operation, or stacked error conditions. This approach has been applied to eliminate errors in commercial software systems, networking, industrial controls, and increasingly in energy applications. We have recently demonstrated the use of standard distribution feeders as a basis for TLA+ models in order to provide a test setup for investigating distributed energy control algorithms. Here we examine a distribution feeder under conditions in which a transmission outage curtails slack bus power flows. While conventional grid architectures under these conditions would de-energize the feeder and require nodes with distributed energy resources (DERs) to operate in islanded mode, we model control algorithms for a transactive energy system in which DERs are able to sell power to neighboring nodes. A modular architecture is used to add new node and feeder capabilities, such as the ability to buy and sell energy in hyperlocal distribution markets, as module upgrades while containing modifications to the control system used to operate the feeder. This approach allows the resiliency benefits of transactive energy to be gained while minimizing implementation costs through the reduction of complexity. We model a laminar coordination framework and use TLA+ to formally verify its operation. Using this formal specification, we investigate the latency of coordination signals over a range of system states and identify conditions for stable operation. We show that while allowing energy transactions between peers on a feeder improves system resilience by permitting continued operation despite the failure of transmission infrastructure, care must be taken to address other failure modes that arise from this decentralized architecture which can be addressed through model checking. This work establishes formal verification as an invaluable tool for realization of the resiliency benefits of transactive energy by uncovering potential failure modes and providing engineers a chance to mitigate them before systems are commissioned.**

Index Terms—**transactive energy, distributed energy resources, formal verification**

I. INTRODUCTION

In the case of a transmission outage, a typical distribution feeder will de-energize even if distributed energy resources (DERs) are present. By contrast, a transactive energy system capable of peer-to-peer (P2P) trades should continue to operate in this situation, with the modification that price-taking DERs become the new price-setting generators within the local distribution system power market.

Several challenges must be overcome to upgrade today's distribution systems in order to realize the benefits of transactive energy. Firstly, the computational requirements for operating transactive markets over a distribution system are immense due to the large number of nodes that must be coordinated and the complex set of behaviors which each node may follow. Illustrating this, there are on the order of 10^8 electric customers in the US and each in principle represents a node that may buy and sell energy in a transactive power system. Each node may plan its behavior according to a complex set of assumptions about its future needs and forecasts regarding future power prices. It will likely be necessary to use a design strategy such as laminar decomposition to reduce complexity and efficiently optimize over this large number of nodes in order to operate local power markets [1]–[3].

A second challenge is that local power markets must be operated according to local rules and preferences. Successfully evolving today's distribution systems towards a future transactive energy grid will require allowing a multiplicity of regional power systems with different rules to interoperate while adopting new technologies and regulations in a piecemeal fashion. This requires a strategy for maintaining comprehensibility and upgradability as systems evolve. We have recently proposed a Functionally Defined Invariant Architecture (FDIA) for the power grid, in which a separation of high-level concerns essential to power system operation acts to reduce interdependencies while allowing upgrades [4], [5].

Thirdly, the challenge of implementing a control architecture capable of integrating all necessary subsystems securely and correctly is significant. A transactive energy system with millions of market participants has a large attack surface in comparison to a traditional power system under centralized control. Additionally, the possibility of errors resulting from improperly integrated components or rare states increases with the number of participating subsystems. Formal verification can help to ensure that integration is performed correctly by both assessing system response to threat models and by evaluating edge cases which may be too rare or extreme to appear during system tests. [5]–[7] Formal verification tools have been increasingly used in industry to ensure safe behavior of critical systems, particularly in the domains of distributed software [8], [9] and IOT systems [10].

In this work, we use a formal model of a transactive energy system employing laminar decomposition within a FDIA to examine a standard IEEE 13-bus feeder immediately before and after a transmission outage. In this scenario, the regulatory module within the FDIA enables a local power market on the distribution feeder and an optimal price is determined based on communication between the top-level coordination domain and sub-domains. We then examine the effect of latency preventing coordination signals from being received during optimization.

II. Formally Verified Transactive Energy System

A. System architecture

In order to allow technologies, regulations and market rules in the power system to be modified, a FDIA was employed [5]. We modeled FDIA modules handling Regulation (RE), Communications (COM) between the control system and nodes, State Estimation (SE), and Optimization (OP) of feeder state. The topology used to model power flows and transactions was a standard IEEE 13-bus feeder system. As discussed in previous work [5], a Python script was used to read in the feeder information from an Open DSS script and encode its hierarchical structure in a TLA+ file. The TLA+ specification is then able to use this structure to estimate power flow through feeder branches.

In contrast with our previous work, we modeled a laminar coordination framework to assess an optimization algorithm able to limit interdependencies between node preferences and achieve improved computational complexity in real-world scenarios. As shown in Fig. 1, the 13-Bus system was divided into three coordination domains. The SE module was modified so as to calculate branch power flows within each coordination domain, using the output of one sub-domain as a node within a larger domain.

An FDIA allows modules implementing high-level tasks to be substituted for each other while maintaining the interfaces between modules, facilitating system upgrades. In the TLA+ script specifying feeder settings, we added a sequence named OP_SlackBusPower which specified a limit to power supplied to the feeder from the transmission system as a function of time. Additionally, we modified the RE module which specifies the rules and settings that all other FDIA modules must follow. With this modification, the RE module switches the OP module between a traditional flat rate optimization procedure and a transactive optimization. The first case, a flat rate OP, simply communicates a single rate determined by the slack bus to all nodes. The second case implements a transactive energy system using a laminar architecture, and optimizes power price as described below.

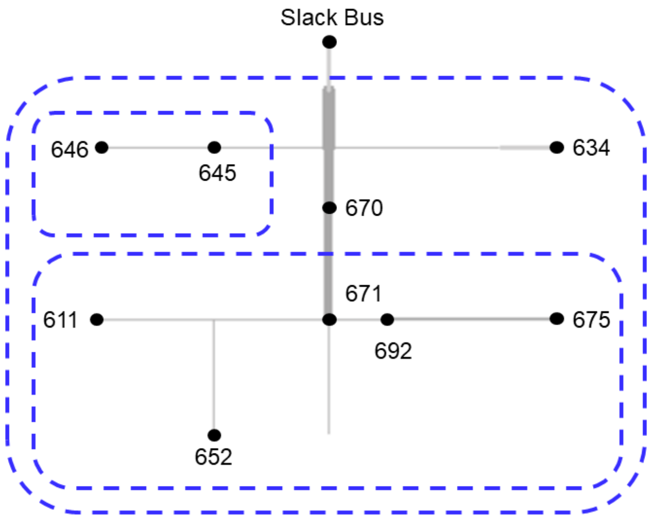


Fig. 1. The standard 13-Bus feeder system used for this analysis. Power consuming or producing nodes are labeled. Three coordination domains were used, corresponding to a top-level coordination domain and two sub-domains. Domains are demarcated with blue dotted lines.

B. Formal model

The TLA+ specification [11] consists of three scripts:

a) FDIA model: determines a formal model of the FDIA. It comprises a set of operators acting to perform computations over the feeder, and a set of processes written in Pluscal and translated into TLA+ code. Processes correspond to initialization of the specification, FDIA modules performing Regulation (RE), State Estimation (SE) and Optimization (OP/OPI), processes representing communications (COM) between the control system and each node in the system, as well as a clock (CL) process which increments the time step.

b) Settings: determines the settings necessary for defining the range of system conditions under evaluation. In particular, this script defines settings relating to optimizing system state when running in transactive mode. These settings include $OP\_tolerance\_kW$, which defines the tolerance necessary for convergence, and $OP\_SlackBusPower$, which is a sequence defining the maximum power available denoted in kW from the transmission system as a function of time step.

c) Feeder structure: describes the hierarchical power system structure and is automatically generated by a python script. This script includes demand curves for all feeder nodes. This approach allows the formal model to be run using realistic parameters corresponding to test feeder states. Coordination domains are specified in this file as a set of branches.

C. Optimization over laminar coordination framework

In the TLA+ specification, nodes are able to determine power output in response to a price signal. In order to introduce distributed generation, we modified the python script producing node demand curves such that negative demand (ie. generation) is introduced at high prices. Implementing the laminar coordination framework then consists of determining the price at which load and demand are balanced within the feeder, by communicating coordination signals between the top-level problem corresponding to slack bus power and sub-problems corresponding to sub-coordination-domains as shown in Fig. 1.

In this specification, we are examining a simple case in which there are no power flow constraints which could result in congestion causing the price to vary between domains. However, we are additionally not assuming that the coordination domains have information about node behavior beyond an ability to query the node with a price and receive a power quantity - representing generation or consumption - in return. Because of these minimal assumptions, it would be possible to implement this optimization algorithm for consideralby more complex node behaviors.

The price of power in each coordination domain is determined using the distributed procedure described in Algorithm 1. Within the TLA+ specification, this is implemented using separate processes for the OP and SE modules along with a COM process representing communication with each node in the feeder system. The while loop is implemented by modifying the $processAvailable$ function, which allows OPI to update prices and signal that another optimization iteration is necessary.

For $OP\_iterations >= 3$ in Algorithm 1, an adaptive step size is used. This step size is determined by the formulas:

$$OP\_remain_j = \ OP\_tolerance\_kW - OP\_netPower_{j-1} \tag{1}$$

$$dP_j = OP\_netPower_{j-1} - OP\_netPower_{j-2} \tag{2}$$

$$stepSize_j = \frac{OP\_remain_j}{2 * dP_j} * stepSize_{j-1} \tag{3}$$

$$\begin{aligned} \textbf{if } & Abs(stepSize_j) < 1 \\ \textbf{then } & stepSize_j = stepSize_j / Abs(stepSize_j) \end{aligned} \tag{4}$$

For iteration j, the change in price $stepSize_j$ is calculated to be half of the ratio of OP_remain_j, the difference between the target and current net power at the slack bus, and dP_j, the difference in power between the previous two iterations. Eq. 4 adds the additional constraint of a minumum step size corresponding to a price change of 1 cent per kWh. In the case that there was no change over the previous two iterations and the net load is outside of tolerance, the step size is doubled as described in algorithm 1.

The $chooseLoad$ operator mentioned in algorithm 1, implemented in the FDIA script, uses demand curves for each load specified in the feeder model to determine a load for a given node based on a price.

Algorithm 1: Price optimization

```
Result: Distributed algorithm determines a price such
        that top-level coordination domain net power
        is within constraints
initialization: $OP\_iteration = 0$;
while $|OP\_netPower| > OP\_tolerance\_kW$ do
    $OP\_iteration_j = OP\_iteration_{j-1} + 1$;
    Calculate $OP\_remain_j$, $dP_j$ as described in the
      text
    if $OP\_iteration_j == 1$ then
        $OP\_rate = OP\_initialPrice$;
    else if $OP\_iteration == 2$ then
        if $dP_j < 0$ then
          $OP\_Rate = OP\_Rate + OP\_initStepSize$;
        else
          $OP\_Rate = OP\_Rate - OP\_initStepSize$;
    else
        if $dP_j == 0$ then  $StepSize = 2 * StepSize$;
        else calculate step size as described in text;
        $OP\_Rate = OP\_Rate + StepSize$;
    Communicate $OP\_rate$ to coordination domains
    for $n$ in $\{node_n\}$ do
        COM: Node receives price from coordination
          domain, determines load using $chooseLoad$
          operator, and communicates load to
          coordination domain;
    end
    SE: Recursively calculate load within each
      coordination domain; Assign load for each
      domain $i$ to $coordDomainPower_i$;
end
```

III. Effect of transmission outage

A rule was added to the RE module in order to switch from a flat rate to a transactive optimization algorithm in response to transmission curtailment. A sequence of two time points was examined, corresponding to points immediately before and after an outage. At time $t = 1$, a flat rate of $\$0.18/kWh$ was communicated to all nodes and each node was able to determine its consumption with effectively no net power constraints. This flat rate resulted in the radial power flow distribution shown in Fig. 2 (A), graphed using a Python script to convert TLA+ output into OpenDSS settings.

At the second time step, power flow into the slack bus is curtailed and the distributed optimization algorithm described above is employed. The system state was determined by the exchange of coordination signals comprising (1) price information travelling down from the top-level domain to sub-domains and nodes, and (2) power information determined by individual nodes then aggregated at successive coordination domain levels. The invariant $time \in timeSteps \lor |coordDomainPower[slackBusBranch] - slackBusDispatch| < OP\_tolerance\_kW$ was added to the TLA+ model. This invariant expressed the condition that upon termination of the algorithm, the net power at the slack bus is within tolerance. The specification terminated with 17,564,111 states (1,573,167 distinct states), indicating that for all reachable states representing permutations of allowed FDIA system transitions, the optimization succeeds. This is consistent with expectations, as the demand curves are monotonic.

An example of algorithm convergence is shown in Table I. This shows a convergence after five iterations, demonstrating an overshoot followed by a correction, and resulting in a net power within the limit specified by $OP\_tolerance$. As shown in Fig. 2 (B), this resulted in a lower absolute amount of power flow with a cocentric peer-to-peer rather than a radial pattern.

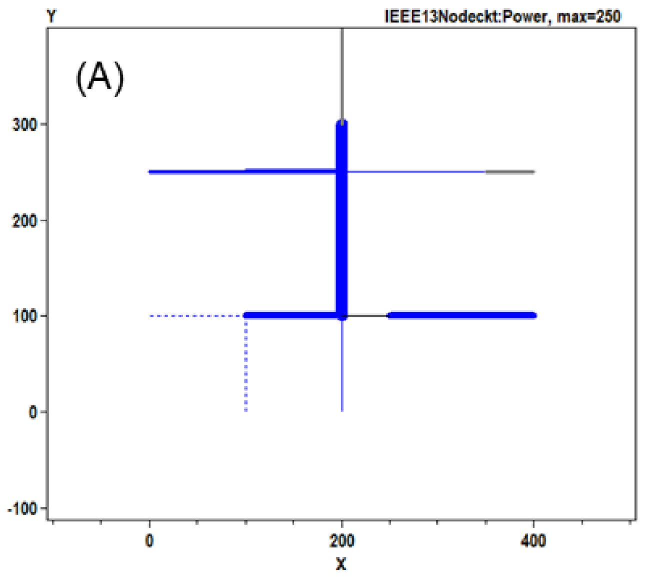
TABLE I
POWER FLOW AND PRICE OPTIMIZATION FOLLOWING OUTAGE

Iteration	Price (USD/kWh)	Net Load (kW)
1	20	1074
2	30	709
3	39	492
4	49	-131
5	47	22

IV. INSTABILITIES DUE TO LATENCY

We next examined the effect of latency on coordination signals traveling from feeder nodes to the control system in which not all node responses arrive on every optimization iteration. In order to model latency, we introduced a $latencyOptions$ function into the system settings file. This function maps node names to a set of possible latencies. We then added a $latencySetup$ label to the RE module, which uses a **with** statement to choose a latency for each node on each timestep. In PlusCal, **with** is a non-deterministic statement generating a parallel timeline corresponding to each possible option. The result is that the specification evaluates a branch corresponding to every available latency combination. Nodes were prevented from responding unless $Mod(OP\_iterations, latency[self]) = 0$ in which Mod corresponds to the modulus operator. Thus, nodes with a latency of 1 respond during every iteration of the optimization algorithm, a latency of 2 indicates a response for every other iteration, etc.

While the node demand curves are monotonic and well-behaved, so that the algorithm will inevitably converge in the case that all coordination signals arrive and a solution within the required tolerance exists, these properties are not robust in the case of signal latency. As a result, infinite loops in the optimization algorithm are possible. In order to detect these loops, we tracked the set of variables necessary to fully determine the state of the optimization algorithm. These variables included the net power corresponding to the curent and previous iterations, the step size, the current price, and the set of nodes skipped during this iteration due to latency. For this latency model, repetition of a set of these variables implies that the system is looping. We added an invariant used to detect these loops, requiring that repetitions of system state



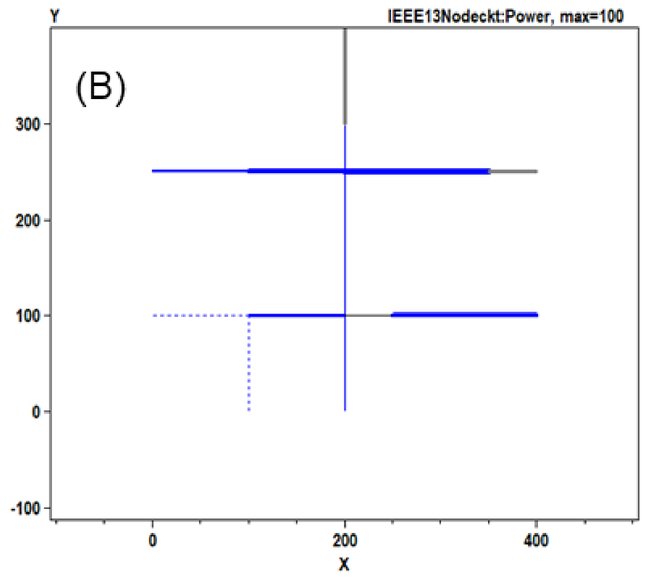


Fig. 2. The effect of a transmission outage, with TLA+ system state output solved using OpenDSS. (A) Power flows through the 13-Bus system under the flat $\$0.18/kWh$ condition. (B) Power flows resulting from a transactive system after transmission outage. The pattern of power flow is clearly distinct from the flat rate case, as the slack bus is curtailed and local generation must balance local load.

variables are not found if the algorithm has not converged. Allowing nodes to adopt combinations of latency values then revealed conditions in which the optimization algorithm failed to converge.

For example, an error trace corresponding to Table II was discovered allowing two nodes to adopt $latency \in \{1, 2\}$. This produced a repetition of system state variables after 11 iterations in this timeline corresponding to a series of 214 states and 54,528,582 surveyed states total in the specification (4,991,366 unique states). In this situation, two nodes experience latency such that they return a signal only every other iteration. This produces a cyclic instability which prevents the algorithm from converging.

Such instabilities are the result of nodes with steps in their demand curve near the equilibrium price, for which these steps are large compared to the feeder tolerance. For example, the error state shown in Table II involves nodes with price discontinuities at $0.42/kWh and $0.45/kWh that are larger than the feeder tolerance of 50 kW. Similar instabilities were discovered through additional analysis. This suggests several strategies for improving system resilience. One strategy is to require that nodes submit demand curves rather than only a response to single-point queries. Similar to present-day security constrained economic dispatch, these bids would allow the optimizer to calculate prices with effectively no latency. A second strategy is to increase the feeder tolerance through, for example, utility-owned DERs. A third strategy is to increase the size of the coordination domains, and rely on customers to have a distribution of preferences that decreases the reliance on any individual node and lowers the relative contribution of any individual demand discontinuity.

TABLE II
LACK OF CONVERGENCE DUE TO LATENCY. UNITS AS IN TABLE I

Iteration	Price	Net Load	Step	Skipped
1	20	846	20	{“646n”, “671n”}
2	30	1581	10	None
3	19	1946	-11	{“646n”, “671n”}
4	48	-462	29	None
5	47	-309	-1	{“646n”, “671n”}
6	45	-120	-2	None
7	44	-120	-1	{“646n”, “671n”}
8	42	512	-2	None
9	43	512	1	{“646n”, “671n”}
10	45	-120	2	None
11	44	-120	-1	{“646n”, “671n”}

V. CONCLUSION

Unlike a distribution system with a typical radial, centralized power flow, a transactive energy system may continue to operate during periods of transmission outage as examined in this work. This is a capital-efficient route to improving system resilience as it leverages DERs that are not owned by the power utility. Furthermore, the use of an FDIA to allow the system to readily switch between a radial and a transactional mode is an example of modular decomposition based on high-level functions enabling flexible power infrastructure affording both the efficiency of centralized power systems and the resilience benefits of the transactional paradigm. However, the optimization required for a transactive system is more complex because it involves coordinating assets under the control of different entities and thus requires coordination signals to be bidirectionally exchanged. To realize improved resilience therefore requires reliable communications as well as correct and secure implementation of distributed control algorithms.

By coupling OpenDSS models with TLA+ specifications via Python scripts, we have shown that formal verification of optimization algorithms for transactive energy is complementary to existing best practices for engineering robust power systems. By examining millions of system configurations we uncover rare edge cases which would be difficult to discern from traditional error analysis and planning techniques, and suggest changes at the algorithm and protocol level to address them. However, bounds on algorithm instability are dependent on system design at the feeder and aggregate level and coupled opportunities for improving system resilience exist at all of these levels. Further formal verification work on subsystem interoperability and cybersecurity is likely to show similar coupled opportunities.

REFERENCES

[1] J. D. Taft. “Grid Architecture: A Core Discipline for Grid Modernization.” IEEE Power and Energy Magazine, pp.18-28, August 2019

[2] J. D. Taft. “Taft, Jeffrey D. Architectural basis for highly distributed transactive power grids: Frameworks, networks, and grid codes.” No. PNNL-25480. Pacific Northwest National Lab.(PNNL), Richland, WA. June, 2016

[3] D. Holmberg, M. Burns, S. Bushby, A. Gopstein, T. McDermott, Y. Tang et al. “NIST Transactive Energy Modeling and Simulation Challenge Phase II Final Report.” NIST Special Publication 1900-603. May, 2019.

[4] A. Ransil, E. F. Fongang, M. Hammersley, I. Celanovic, F. O’Sullivan. “A Computable Multilayer System Stack for Future-Proof Interoperability.” [Conference Presentation]. IEEE PES Transactive Energy Systems Conference (TESC). July 2019.

[5] A. Ransil, M. Hammersley, F. O’Sullivan. ’“Modularize, but Verify.” Manuscript Under Review.

[6] S. Gayathri, R. Selvamuthukumaran, M. Novak,T. Dragicevic. “Supervisory energy-management systems for microgrids: Modeling and formal verification.” IEEE Industrial Electronics Magazine. March 2019.

[7] S. A. Naseem, R. Uddin, O. Hasan, D. E. Fawzy. “Probabilistic formal verification of communication network-based fault detection, isolation and service restoration system in smart grid.” Journal of Applied Logics—IFCoLog Journal of Logics and their Applications 5.1. February 2018.

[8] A. Souri, N. J. Navimipour, A. M. Rahmani. “Formal verification approaches and standards in cloud computing: A comprehensive and systematic review.” Computer Standards & Interfaces. December 2017.

[9] Y.M. Kim, M. Kang. “Formal Verification of SDN-Based Firewalls by Using TLA+.” IEEE Access. March 2020.

[10] S. Latif, A. Rehman, N. A. Zafar. “Modeling of sewerage system linking UML, Automata and TLA+.” 2018 International Conference on Computing, Electronic and Electrical Engineering. Nov. 2018.

[11] A. Ransil. redransil/TLA-laminar: Release v0.14.0 (Version v0.14.0). Zenodo. http://doi.org/10.5281/zenodo.4162501 Oct. 2020.