Synthesizing Voltage Ride-Through Controllers for Data Centers

Wayne Wang
wswang@umich.edu
University of Michigan
Ann Arbor, MI, USA

Archit Bhatnagar
architb@umich.edu
University of Michigan
Ann Arbor, MI, USA

Tongyuan Miao
tymiao@umich.edu
University of Michigan
Ann Arbor, MI, USA

Saniya Kalamkar
skalamka@umich.edu
University of Michigan
Ann Arbor, MI, USA

Wenqi Cui
wenqicui@nyu.edu
New York University
New York, NY, USA

Inigo Incer
iir@umich.edu
University of Michigan
Ann Arbor, MI, USA

Ang Chen
chenang@umich.edu
University of Michigan
Ann Arbor, MI, USA

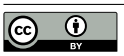
Abstract

Data centers are among the power grid's fastest-growing loads. Since data center servers are sensitive electronic components, they need to be protected against the grid's voltage disturbances during grid faults. While this can be achieved by simply disconnecting from the grid upon disturbances, disconnection can further destabilize the power system if many data centers trip at once. To address this emerging concern, voltage ride-through (VRT) grid codes have been proposed to standardize data center behavior. They require a data center to stay connected for a period of time through the disturbance, hold an active power floor, and recover its draw within a deadline upon restoration. However, systematically designing and certifying controllers that satisfy these coupled temporal and operational requirements remains challenging.

We propose SolVRT, a system that synthesizes a grid-code-compliant VRT controller for a given data center using formal methods. We develop a specification language that expresses a grid code in Signal Temporal Logic (STL) as the basis for formal reasoning. Our encoding algorithm takes the specification, along with a model of the data center's power topology, and translates the constraints into a controller synthesis problem. This step produces a correct-by-construction controller if a solution can be found, or a proof that no such controller exists. For the latter case, SolVRT provides a diagnostic step: it traces the facility's "conflict frontier," isolates the conflicting clauses that led to non-compliance, and computes the smallest hardware or workload change that would enable compliance. We evaluate SolVRT through closed-loop simulations of a 200 MW data center connected to a 140-bus transmission system. The results demonstrate that SolVRT can synthesize compliant VRT controllers, certify infeasibility when compliance is unattainable, and identify targeted modifications that enable compliance.

CCS Concepts

• **Software and its engineering → Formal methods**; *Specification languages*; • **Theory of computation → Modal and temporal logics**; • **Hardware → Enterprise level and data centers power issues**; *Power networks.*



This work is licensed under a Creative Commons Attribution 4.0 International License.
ACM DXConf 2026, Ann Arbor, MI, USA
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-3028-3/2026/10
https://doi.org/10.1145/3847238.3849043

Keywords

Data centers, voltage ride-through, signal temporal logic, controller synthesis, formal methods

ACM Reference Format:
Wayne Wang, Archit Bhatnagar, Tongyuan Miao, Saniya Kalamkar, Wenqi Cui, Inigo Incer, and Ang Chen. 2026. Synthesizing Voltage Ride-Through Controllers for Data Centers. In *The Inaugural ACM Conference on Digital Transformation (ACM DXConf 2026), October 29–30, 2026, Ann Arbor, MI, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3847238.3849043

1 Introduction

Data centers (DCs) are large electric loads with fluctuating demands and high sensitivity to power quality; their response to grid voltage disturbances has been a rising concern. For instance, if the grid experiences voltage sags upon contingencies and faults, data centers may simply disconnect from the grid in order to protect internal devices (e.g., by switching to local power sources such as uninterruptible power supplies, or UPSes, and diesel generators). From the grid's perspective, however, simultaneous disconnection of large loads introduces further disturbances during an already-stressed event. In a 2024 incident in Virginia, a single transmission-line fault caused voltage sags that tripped roughly 1,500 MW of data center load at once, most of which stayed offline for hours [32]. This has prompted the development of voltage ride-through (VRT) requirements from grid operators, such as ERCOT (US) and Energinet (Europe). VRT requires the data center to remain connected and follow specified response criteria if the grid voltage sags or swells during a transient fault.

While VRT has been well studied for power devices and generators, data centers present a new range of challenges. First, DCs represent a large *load*, whereas VRT standards were originally designed for *generators*. Further, DCs comprise a *system of devices*, each with its own protective response mechanisms invisible to the grid, requiring system-level joint control. The growth of *AI workloads* compounds these challenges: a single data center campus now draws hundreds of megawatts, its training workload swings within seconds, and the hardware at risk (e.g., GPU clusters) is costly enough that individual protections are tuned to trip early.

Today, a data center's response to a voltage event heavily depends on individual device protection settings, power electronics vendor behavior, and controller heuristics. Under this uncertainty, DC operators default to early disconnection to protect their internal operation. A hand-tuned controller might help the DC ride through, but provides no compliance guarantee since grid codes

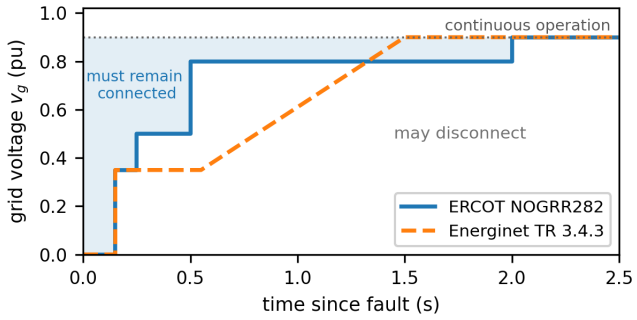


Figure 1: Low-voltage ride-through envelopes of the two codes: ERCOT NOGRR282's band envelope (May 2026) and Energinet TR 3.4.3's voltage–time curve.

impose several timed requirements that the data center must satisfy simultaneously. A principled design for the data center to control internal devices and ride through voltage events with guaranteed grid code compliance is still missing. This difficulty is compounded by the fact that the grid codes exist as natural language documents describing numeric constraints, which are hard to analyze formally in a machine-checkable manner. This interface between grid requirements and data center capabilities is described in informal text, making it difficult for grid operators to anticipate data center behavior, for data centers to guarantee compliance, or to reason about the hardware or workload provisioning that would achieve compliance under all conditions.

In this paper, we take a fresh look at VRT from the perspective of formal methods. If we can formalize grid codes in a machine-checkable language, a range of synthesis and verification methods becomes available to attack this problem. We argue for treating VRT requirements as a formal assume-guarantee specification, a method widely used in safety-critical domains such as aerospace engineering [21, 36]. With the assume-guarantee interface, the grid states assumptions on the voltage disturbance, and the data center must uphold guarantees on its response over explicit signals whenever those assumptions hold. Both assumptions and guarantees are timed constraints on real-valued signals, and Signal Temporal Logic (STL) expresses exactly such predicates over signal trajectories [29]. Compliance can be established by solving whether a controller exists that enables the data center to satisfy the guarantee under the assumption while respecting all constraints.

Following this intuition, we propose a grid-code-aware system SolVRT that solves the VRT compliance problem for data centers using formal methods by synthesizing a code-compliant controller for a given data center. When this synthesis is infeasible, SolVRT diagnoses the unsatisfiable clauses and provides change recommendations. We make three contributions. First, we develop a new domain-specific language (DSL) for expressing data center VRT grid codes. The specification makes voltage event assumptions, response requirements, timers, active-power floor, and internal protection limits explicit and machine-checkable—using STL predicates to capture the core requirements. Next, our algorithm encodes the specification into a controller synthesis problem with other constraints, including disturbance traces, plant models, and hardware constraints. When the synthesis query is feasible, SolVRT outputs a model predictive controller (MPC) as witness. If synthesis is infeasible, the diagnosis and provisioning step takes over. We diagnose the violating clauses and provide provisioning recommendations by modeling infeasibility as the *conflict frontier*. This allows SolVRT to identify the unsatisfiable core of the conflict and compute the minimal changes needed on hardware capability or workload flexibility to restore compliance for a given data center and specification.

With a comprehensive simulation using a 200 MW data center and a 140-bus transmission system, we are the first to formalize VRT requirements for data centers as a controller synthesis problem and demonstrate its practicality. We hope this paper will inspire future work that studies DC/grid coordination through the lens of formal analysis to increase the resilience of cross-infrastructure interactions. We will release SolVRT in open source.

2 Background and Related Work

2.1 What are VRT and LVRT

Voltage ride-through (VRT) has traditionally been defined as a generator-side requirement [20, 38]. Its central objective is to ensure that generators, including synchronous machines and inverter-based resources such as solar photovoltaic and wind generation, remain connected during short-duration voltage disturbances caused by grid-side faults. By preventing premature disconnection, VRT requirements help avoid cascading loss of generation and support system stability during and after fault events.

Comparable ride-through requirements have not been imposed on loads, which were typically smaller, more dispersed, and had a less explicitly regulated dynamic voltage response. However, the rapid growth of large electronic loads, particularly DCs, is changing this assumption. Since a single DC today can represent a substantial amount of load, its disconnection during a voltage disturbance can create an abrupt loss of demand and potentially worsen grid dynamics. As a result, large-load ride-through capability is emerging as an important reliability requirement. For example, ERCOT [16] is developing large computational/electronic load ride-through rules through NOGRR282/NPRR1308, motivated by observed DC and cryptomining load trips during normal voltage disturbances. When data centers are clustered in a region, the grid-side impact is aggravated. In the 2024 Virginia incident [32], a single localized transmission fault propagated to data-center facilities across the region as a voltage sag, triggering their independent protection at once and amplifying the fault into a total load loss of roughly 1,500 MW.

These emerging codes share a three-part structure: the DC must stay connected while the voltage disturbance lies within a ride-through envelope (Figure 1), keep its active power draw above a floor during the sag, and restore its draw within a deadline after the sag clears. Codes differ only in the values that each part takes. For example, Energinet's VRT code [15]—the Danish code for transmission-connected demand facilities—places requirements on the same structure of three obligations as the ERCOT code, and the two codes only differ in the requirement values. Regarding how long the facility must stay connected during voltage sags or swells, ERCOT specifies voltage ranges with corresponding ride-through durations; in contrast, Energinet requires a piecewise-linear voltage–time curve. Regarding active power that the facility must keep drawing during the fault, ERCOT requires a level proportional to the sag for certain voltage ranges, while Energinet permits a floor of zero for all depths. Regarding recovery of active power draw

after the fault, ERCOT sets a single target of 90% of the pre-fault draw within two seconds, whereas Energinet stages three targets over 30 seconds (80/90/95% at 5/20/30 s) and caps the overshoot.

VRT grid codes may require data centers to ride through both voltage sags and swells, but sags are the more challenging case. Upon sags, low-voltage ride-through (LVRT) involves all three requirements (connection, minimum power draw, recovery); but high-voltage ride-through (HVRT), in the example of ERCOT, imposes a connection requirement and an unchanged active power draw, making it an easier subset of the LVRT problem.

Hence, we focus on LVRT in this paper, although SolVRT's grammar can encode voltage swell and related power draw requirements as well and the synthesis query stays the same. Unlike conventional LVRT requirements that are often specified for an individual generator or power-electronic device, data-center LVRT must be understood as a *system-level property*. A data center consists of multiple interacting components, including UPS systems, battery energy storage systems (BESS), switchgear, power-electronics interfaces, internal distribution networks, and workload-dependent power consumption. In practice, during a voltage sag, tripping is initiated not by the computing equipment itself but by facility protection tuned to protect it. In the 2024 Virginia event, UPS control schemes that count voltage disturbances tripped the facility after repeated sags [32]. LVRT analysis and certification for data centers is therefore fundamentally different from conventional device-level verification: it requires certifying that the aggregated data-center system satisfies grid-side ride-through requirements while respecting internal device constraints.

2.2 Data Center Electrical and Compute Models

From an electrical perspective, prior work models DC power delivery using UPS systems, BESS, switchgear, power-electronics converters, and internal distribution topology [44]. These components make the load behavior different from a passive constant-power load through a combination of protection settings, converter limits, and reactive-power capability, which affects the data center's capability to comply with LVRT grid codes during a voltage disturbance. We adopt LinDistFlow [5, 48], a standard linearized power-flow approximation for radial distribution networks, to model the internal network of the DC. Specifically, the DC network extends from the grid interconnection point through transformers and distribution lines to internal buses, where compute loads and UPS and BESS converters are connected. UPS and BESS resources are especially relevant for LVRT because they sit between the grid and the computing load [3, 41]. They can buffer active power, provide or consume reactive power, and protect internal equipment from voltage disturbances. Their behavior is constrained by hardware ratings: active-power limits, reactive-power limits, energy limits, and apparent-power capacity. In power-electronics models, active power P and reactive power Q often share the same converter capacity S, e.g., $P^2 + Q^2 \leq S^2$, as in recent work on data center ride-through and UPS control [31, 40]. These converter resources are typically what prior ride-through controller designs actuate: centralized and decentralized voltage control over the facility's internal distribution network [44], grid-forming control of UPS and BESS converters [1, 37], passivity-based converter control for DC facilities [42], and coordinated grid-side storage that absorbs load transients [26, 30]. These designs are validated on simulated faults, but treat the grid code informally: they can neither certify compliance nor establish that no compliant controller exists.

In terms of compute, data centers are increasingly dynamic loads. Prior work on GPU and AI workload power management shows that software can change the power profile through power caps, frequency scaling, scheduling, and workload shifting, for efficiency and demand response [2, 12, 28, 43]. A related line coordinates data centers with the grid in normal operation, including voltage regulation via GPU-level actuation [27]. These software knobs act on timescales too coarse for a low-voltage event, so they cannot serve as fault-time controls. The workload still impacts the ride-through problem through the pre-fault active power, which sets the grid code's power floor and recovery target.

2.3 The Case for LVRT Formal Specifications

Formal specifications are logical statements with precise semantics, in stark contrast to natural language, which is the common method for describing grid codes. Their precision is what enables automated verification and synthesis, and safety-critical domains have long adopted them for exactly this reason. Specifically, assume-guarantee specifications have been proposed as the algebra of compositional system design [9, 21, 22]: one side states assumptions on its environment, and the other upholds guarantees whenever those assumptions hold. They have been applied to aircraft electric power systems [33], space missions [36], and networked control [10, 23]; this paper argues for taking a similar approach to LVRT analysis.

In particular, temporal logics support the expression of statements over signals that change over time. Within the class of temporal logics, Linear Temporal Logic (LTL) and Metric Temporal Logic (MTL) operate over boolean propositions, while LVRT grid codes constrain real-valued signals against numeric thresholds under deadlines. Signal Temporal Logic (STL) specifies requirements over exactly such real-valued signals in continuous time [29]. Its atomic predicate is any Boolean-valued statement μ over the signals x being considered. Its formulas combine predicates with Boolean operators and time-bounded temporal operators,

$$\varphi ::= \mu(x) \mid \neg\varphi \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \Box_{[a,b]}\, \psi \mid \Diamond_{[a,b]}\, \psi \mid \varphi\, \mathcal{U}_{[a,b]}\, \psi, \quad (1)$$

where $\Box_{[a,b]}$ reads "always on $[a, b]$," $\Diamond_{[a,b]}$ reads "eventually within $[a, b]$," and $\varphi\, \mathcal{U}_{[a,b]}\, \psi$ reads "φ until ψ within $[a, b]$". Beyond the boolean verdict $x \models \varphi$, STL has a quantitative *robustness* $\rho(\varphi, x)$: positive exactly when x satisfies φ, with its magnitude being the margin of satisfaction or violation [6, 17]. These operators can express many cyber-physical requirements, such as maintaining a signal within a safe range, responding within a deadline, or satisfying a condition over a time interval. LVRT has this same structure, since voltage ride-through, active power floor, and recovery are all time-dependent. Synthesis from temporal-logic specifications, which decides whether any controller can satisfy a given specification, is studied extensively in robotics and control [7, 25]. For STL in particular, synthesis can be formulated as an optimization problem [34, 35]: Given a plant model $y = g(u, w)$ mapping a control signal u and a disturbance w to the output y, a feasible control set

symbol	meaning
v_g	grid voltage at the point of connection
v_i	min internal bus voltage over n compute buses
p	aggregate active power drawn from the grid
p_{pre}	pre-fault active power (i.e., p at the time of the fault)

Table 1: Signals used to encode LVRT grid code. Voltages are in per unit (pu) of nominal.

K, and an STL formula φ, synthesis solves

$$u^\star = \arg\min_u \; J(u) \quad \text{s.t. } \rho(\varphi, y) \geq 0, \quad y = g(u, w), \quad u \in K, \quad (2)$$

for a control cost J. The constraint $\rho \geq 0$ requires even the tightest clause of φ at its tightest instant to hold. BluSTL [14] implements this method by encoding the bounded STL formulas as mixed-integer constraints over a discretized time window, solving (2) inside a receding-horizon loop and applying the first action each step, yielding a model predictive controller (MPC). For power systems, this synthesis approach has targeted generation-side frequency regulation with energy storage [45, 46] and grid-supportive distributed energy resource modes [47], but not load-side ride-through. We build our synthesis on a Python reimplementation of BluSTL and instantiate (2) for our plant and specification in §3.3.

3 SoLVRT Design

LVRT grid codes require data centers to stay connected during low voltage events. To do so, a data center must carefully control its energy storage resources and UPS systems. This section investigates how the data center should perform this control. We first formalize grid codes using STL, which expresses LVRT requirements in a machine-readable format. We then provide a synthesis algorithm to find a controller for a given data center, deriving a control law and certifying its compliance with the grid code. If no such control law can be found, we perform diagnosis to suggest changes to the data center's parameters to achieve compliance—e.g., by increasing power electronic device capacity, redesigning the data center topology, or shaping workload power consumption.

3.1 LVRT Grid Codes

Grid operators have introduced LVRT grid codes, which regulate the behavior of a data center during a voltage event and usually contain three components. The grid codes require the data center to stay connected during a grid voltage sag, keep active power draw above a level during the sag, and recover active power draw within a deadline after the sag is cleared. The following simplified example is provided by ERCOT [16]:

ERCOT LVRT grid code snippet

A data center shall remain connected to the Transmission Grid during voltage conditions requiring ride-through. When grid voltage falls below 0.9 pu but remains above 0.8 pu, the data center shall continue consuming active power during the low voltage condition, and may reduce it proportional to the voltage drop, but shall return to 90% of its pre-disturbance level within two seconds of voltage returning above 0.9 pu.

LVRT involves the following three signals. *(i)* v_i: In order for a data center to stay connected, its internal compute bus voltage v_i must be above a threshold; otherwise, overly low voltages will trigger protection mechanisms that disconnect the data center from the grid and switch to a backup power source (e.g., diesel generators), in order to prevent damage to power electronics inside the data center. *(ii)* p: The data center draws an aggregate active power p from the grid; at the time of the fault, the power draw p_{pre} is called the pre-fault active power, read off the workload trace at the fault instant. This is the reference value, and it sets the scale of the power floor during the fault and the recovery target during recovery. *(iii)* v_g: The grid voltage v_g at the point of data center / grid connection is the disturbance: this signal sags during a grid fault, and triggers an LVRT response from the data center. Figure 2 shows where these three signals are measured in the data center. The duration for which the data center is required to stay connected to the grid depends on the severity of the sag. For more severe sags, the required duration is shorter; if the sag exceeds the threshold, the data center is allowed to instantaneously disconnect, and the grid code requirements are lifted. Table 1 lists the signals.

As §2 describes, LVRT codes share the same three obligations and differ only in the values each takes. The next subsection introduces a language that the grid code author can use to turn these requirements into formal specifications.

3.2 Formalizing LVRT Grid Codes

As discussed in §3.1, the LVRT requirement is defined over multiple post-fault time segments, with time-coupled conditions and nontrivial rules for active-power floors during voltage sags and recovery. This temporal coupling and piecewise structure make it challenging to directly verify data center compliance using static operating-point analysis or conventional device-level checks. STL, on the other hand, has enough expressivity to write these requirements. Specifically, through the *always* (□) and *eventually* (◇) operators over time intervals, it provides a convenient basis for the formalization of the LVRT requirements. Encoding LVRT requirements in STL not only removes ambiguity, but also enables automated synthesis of a controller for a given data center and grid code. Such a synthesized controller is *guaranteed* to satisfy the requirements under all possible scenarios when such a controller exists. Next, we use the ERCOT example to illustrate this encoding in a new specification language for LVRT grid codes. Figure 3 defines the language, and the rest of this subsection instantiates it for the ERCOT code.

A specification begins with variable declarations. Signals are time-varying quantities measured at the points shown in Figure 2 (e.g., grid voltage v_g); parameters are constant values for a data center (e.g., the equipment trip level v_{trip}); variables are computed from signals and parameters. ERCOT declarations are as follows.

```
signal Vg, Vi, p;  // definitions in Table 1
param  Vtrip;      // equipment tripping voltage
param  T_hold;     // recovery hold window
var    p_pre;      // power draw at time of fault
```

Among these, p_{pre} and T_{hold} are interpretations we make where the ERCOT text is underspecified, rather than values taken literally from it; §5 details both.

An *event* is defined when a signal of interest crosses a threshold. For instance, in the code below, a `fault` event occurs when v_g falls

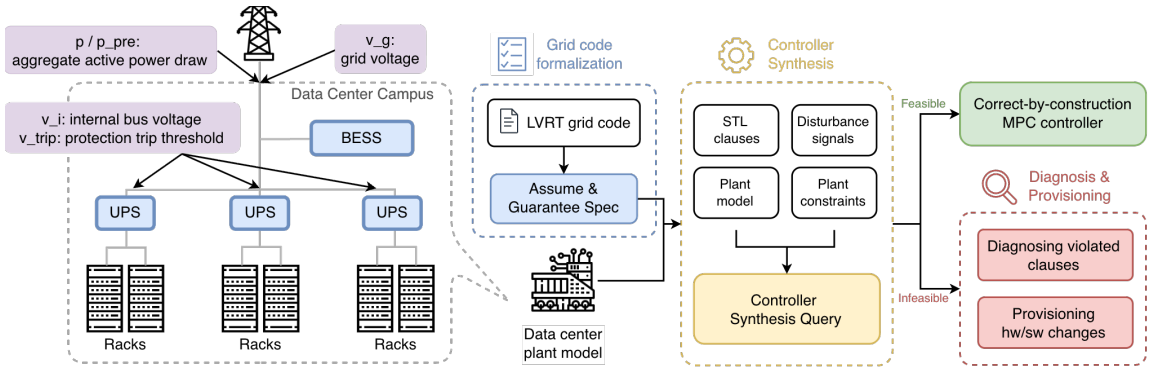


Figure 2: High-level architecture of SoLVRT, comprising three main components: grid code formalization (§3.2), controller synthesis (§3.3), and diagnosis and provisioning (§3.4). A grid code is encoded as a formal spec in our STL-based DSL. The spec and the data center's parameters are encoded into a controller synthesis query. When feasible, the query returns a correct-by-construction MPC controller; when infeasible, diagnosis and provisioning computes changes that restore compliance. A simplified internal power topology of a data center is shown on the left, with LVRT signals at their measurement points.

below the required 0.90 pu, and a `recovery` event occurs when v_g is brought back after a `fault`. The pre-fault power draw is taken from p right before the fault event.

```
event fault     = fall(Vg, 0.90);
event recovery  = rise(Vg, 0.90) after fault ;
p_pre  = p at fault; // sample just before fault
```

§3.1 describes the grid voltage sag ranges and the maximum durations for which the data center must stay connected. These are the conditions under which a data center is required to perform LVRT. The `assume` block captures them as follows:

```
assume {
  var duration = {     // ride-through envelope
    Vg in [0.80, 0.90) => 2.0s;
    Vg in [0.50, 0.80) => 0.5s;
    Vg in [0.35, 0.50) => 0.25s;
    Vg in [0.00, 0.35) => 0.15s;
  }
  recovery - fault <= duration
}
```

In the `assume` block, the ride-through envelope maps each sag depth to its maximum sag duration, and the final line assumes the grid voltage recovers within that duration.

If these assumptions hold, the data center is to uphold three LVRT guarantees shown in the code below. The `guarantee` block first defines the active power floor level μ that the grid code requires, then states the three clauses expressed in STL operators and anchored at the two events, `fault` and `recovery`.

```
guarantee {
  // active power floor
  var mu = {
    Vg in [0.50, 0.90) => Vg * p_pre;
    Vg in [0.00, 0.50) => 0;
  }
  // clause 1: data center stays connected:
  Always[fault, recovery] (Vi >= Vtrip);
  // clause 2: holds power floor:
  Always[fault, recovery] (p >= mu);
  // clause 3: power draw eventually recovers
  Eventually[recovery, recovery+2s](Always[0, T_hold] (p >= 0.9 * p_pre));
}
```

The full requirement is the conjunction of three STL clauses, each a formula φ of the grammar (Figure 3) with its time windows anchored at the declared events. We refer to them as connection (φ_{conn}), active power floor (φ_{flr}), and recovery (φ_{rec}):

$$\varphi := \varphi_{\text{conn}} \wedge \varphi_{\text{flr}} \wedge \varphi_{\text{rec}}. \tag{3}$$

Stay connected. The data center's internal voltage v_i must stay above the power-electronics trip threshold v_{trip} over the fault window $[t_f, t_r]$:

$$\varphi_{\text{conn}} := \square_{[t_f, t_r]}\,(v_i \geq v_{\text{trip}}). \tag{4}$$

Hold the power floor. Throughout the fault duration, the active power must not fall below the floor level μ defined in the `guarantee` block:

$$\varphi_{\text{flr}} := \square_{[t_f, t_r]}\,(p \geq \mu). \tag{5}$$

Recovery. After the recovery event, the active power must reach a target lower bound, $l = 0.9 \cdot p_{\text{pre}}$, within a deadline $T = 2$ seconds, and then hold that level for a window T_{hold}. The *eventually* operator $\Diamond$ over $[t_r, t_r + T]$, with a nested *always* operator $\square$ over $[0, T_{\text{hold}}]$, captures this:

$$\varphi_{\text{rec}} := \Diamond_{[t_r, t_r+T]}\,\square_{[0, T_{\text{hold}}]}\,(p \geq l). \tag{6}$$

The code's text requires only the return of active power within the deadline after grid voltage recovers. We additionally require the level to hold for T_{hold}, so that a momentary spike in power draw does not count as recovery; §5 discusses this interpretation in detail.

On the other hand, if the grid voltage goes beyond the envelope, the assumptions are violated and the data center does not need to perform LVRT.

3.3 Synthesizing LVRT Controllers

Given an LVRT specification, our goal is to determine whether a data center can satisfy it. We address this task using controller synthesis. The data center exposes two signals, the internal bus voltage v_i and the aggregate active power p. Neither can be controlled directly, as they are derived from the active and reactive power (P and Q) that the BESS/UPS devices inject. We express the two converter control actions together as $u := (P, Q)$, which is the control input for the data center.

To characterize how the control input u affects the LVRT capability of data centers, we require a plant model of the data center's internal electrical network. We adopt the LinDistFlow model (§2), under which the internal bus voltage v_i is an affine function

$$
\begin{array}{lcl}
\textit{spec} & ::= & \textit{decl}^{+} \quad \textit{assume} \quad \textit{guarantee} \\
\textit{decl} & ::= & \texttt{signal}\ x \mid \texttt{param}\ x \\
 & \mid & \texttt{var}\ x\ [\,=\ \textit{expr}\,] \\
 & \mid & \texttt{event}\ e\ =\ (\texttt{fall} \mid \texttt{rise})\ (x, \textit{const})\ [\texttt{after}\ e] \\
\textit{assume} & ::= & \texttt{assume}\ \{\ (\textit{decl} \mid \textit{pred})^{+}\ \} \\
\textit{guarantee} & ::= & \texttt{guarantee}\ \{\ (\textit{decl} \mid \varphi)^{+}\ \} \\
\varphi & ::= & \textit{pred} \mid \texttt{Always}[I]\ \varphi \mid \texttt{Eventually}[I]\ \varphi \\
I & ::= & [\tau,\ \tau] \qquad \tau ::= e \mid \tau + \textit{dur} \mid \textit{dur} \\
\textit{pred} & ::= & \textit{expr}\ (\geq \mid \leq \mid <)\ \textit{expr} \mid \textit{expr}\ \texttt{in}\ [\,\textit{const},\ \textit{const}\,) \\
\textit{expr} & ::= & \textit{const} \mid x \mid \tau \mid x\ \texttt{at}\ e \mid \textit{expr}\ (+ \mid - \mid \cdot)\ \textit{expr} \\
 & \mid & \{\ (\textit{pred} \Rightarrow \textit{expr})^{+}\ \}
\end{array}
$$

Figure 3: Grammar of the grid code specification language. Keywords are in bold; x stands for declared names, e for events. signals are measured and params data-center-supplied, while vars are derived; an event binds the instant of a threshold crossing. The assume block describes the admissible disturbance; the guarantee block is a conjunction of clauses φ: STL predicates over time windows I. A predicate is a comparison or a band-membership test.

of the grid-side voltage and the converter injections $u := (P, Q)$. The active power P contributes directly to the aggregate active power p in the LVRT specification, while both P and reactive power Q affect internal voltage. These two actions are coupled by the converter apparent-power rating S, which imposes the constraint $P^2 + Q^2 \leq S^2$. Thus, the reactive power used to support voltage reduces the remaining active-power headroom available to satisfy the active-power floor, and a converter with insufficient capacity may be unable to satisfy both requirements simultaneously. We denote these rating limits as the feasible control set K.

The controller synthesis procedure searches for control actions whose resulting v_i and p satisfy the specification in §3.2, or certifies that none exists. A grid code like ERCOT's expresses the grid-voltage assumption as a set of bands, each paired with a maximum ride-through duration. The assume block admits infinitely many disturbance traces, one for every sag shape inside the envelope, and the controller must satisfy the guarantees under all of them. Therefore, we synthesize against the single worst-case trace v_g' that lower-bounds them all, holding each band at its lowest voltage for its full duration and ordering the bands from deepest to shallowest,

$$v_g'(t) \;=\; \min\{\, v : \mathrm{dur}(v) \geq t \,\}, \tag{7}$$

where $\mathrm{dur}(v)$ is the maximum ride-through duration the specification's assume block assigns to a sag of depth v. The result is the longest and deepest sag the assumption admits, the binding case for the connection clause. It is also the binding case for the connection and active power floor clauses together, although the floor $\mu = v_g \cdot p_{\mathrm{pre}}$ alone is harder to meet at shallow sags. At nominal voltage ($v_g = 1.0$ pu), $v_g \cdot p_{\mathrm{pre}}$ equals p_{pre}, which the data center already sustains. As the sag deepens, μ decreases linearly, while the remaining active-power headroom decreases at a growing rate: holding v_i requires more Q, and under $P^2 + Q^2 \leq S^2$ each additional unit of Q removes more P than the previous one. Thus, since the headroom covers μ at nominal voltage, if it also covers μ at the deepest sag where the floor applies, it covers μ at every depth in between. Because the plant model has no internal state, checking depths covers every trace the envelope admits. The resulting trace is the lower boundary of the ERCOT region in Figure 1; when a code instead specifies a voltage–time curve, as Energinet does (the dashed curve in Figure 1), we use that curve directly as v_g' [15].

The synthesis problem takes two disturbance signals, inputs the controller observes but does not choose: the grid voltage, for which we use the worst-case trace v_g', and the workload's active power draw w_{comp}, the compute demand the facility must serve. We write the two disturbances together as $w = (v_g',\ w_{\mathrm{comp}})$. With the disturbances and the plant model in place, we instantiate the clauses in the specification. The voltage trace sets the fault and recovery times t_f and t_r, turning each time window into a numeric interval. The workload trace sets the pre-fault draw $p_{\mathrm{pre}} = p(t_f)$, turning the active power floor μ and recovery target into numeric bounds. The plant model replaces the output signals v_i and p with their expressions in the control u and the disturbance w. Substituting all three into the clauses of (3) completes the instantiation, producing the concrete STL formula φ over the control variables. Algorithm 1 in Appendix A lists the full procedure.

With the clauses φ, the plant model mapping g, the feasible set K, and the disturbance w all concrete, we instantiate the synthesis problem (2) of §2, with output $y := (v_i, p)$ and cost $J(u) = \|u\|$, selecting the least-effort compliant control $u^\star$ composed of per-step P and Q. We solve the instantiated problem with our Python reimplementation of BluSTL (§2). A feasible query yields an MPC controller that satisfies the instantiated guarantees for the specified voltage trace, workload trace, plant model, and hardware limits. An infeasible query means that no such controller exists for that synthesis instance, which we consider next.

3.4 Diagnosis and Provisioning

The control synthesis query in §3.3 yields a binary result about the feasibility of synthesizing a control law that satisfies the grid code. In cases where such a controller does not exist, we further develop methods to diagnose the clauses of the grid code that cannot be met and compute the changes that could restore feasibility. We obtain both from the same synthesis query by treating the voltage sag depth and active-power floor as parameters of the specification and computing, for each sag depth, the largest floor at which the query stays feasible. We call the resulting boundary the *conflict frontier*, a new abstraction that we use for diagnosis, as shown in Figure 4. This is the maximum active power the data center can guarantee to draw from the grid while staying connected as a function of sag depth. This abstraction allows us to diagnose which grid code clauses cannot be jointly met. Afterward, we modify the data center's parameters until the requirement lies fully inside the feasible region under the frontier.

The conflict frontier. The frontier slopes downward because the connection clause and the active power floor clause draw on the same resource. To maintain the connection clause during a deeper voltage sag, the UPS's grid-facing converter needs to reserve a larger portion of its apparent-power rating for reactive-power support Q to maintain the data center internal voltage v_i. Under the converter rating constraint $P^2 + Q^2 \leq S^2$, this increased reactive-power provision reduces the remaining capacity available for active-power P. Consequently, the largest floor the converter can sustain decreases as the sag depth increases.

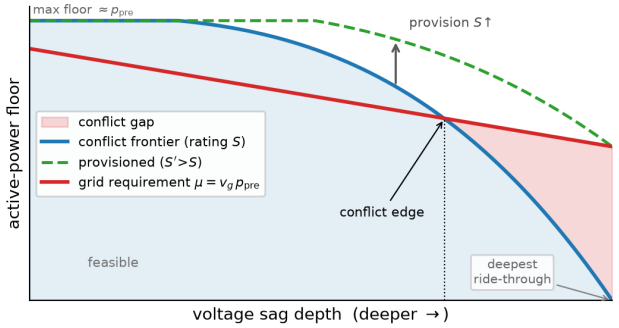


Figure 4: The *conflict frontier*. The frontier is the largest active power floor the data center can sustain while connected at each sag depth. The grid requirement $\mu = v_g\, p_{\text{pre}}$ is the code's floor. Left of their crossing, the *conflict edge*, the requirement is feasible; right of it the requirement exceeds the capability and no controller exists. Provisioning raises the frontier until the requirement lies inside the feasible region.

Computing the frontier. The active power floor for each voltage sag depth is monotone: a lower floor is always easier to sustain since it requires lower P and frees the converter headroom for Q. Leveraging monotonicity, we locate the largest sustainable active power floor for each voltage sag depth using binary search. Sweeping the voltage sag depth traces the frontier, the largest sustainable floor at each depth. The grid code's requirement is the second curve over the same axes, where the code demands the active power floor $\mu = v_g \cdot p_{\text{pre}}$ at each depth. When the requirement exceeds the frontier, no controller exists, and the crossing marks the *conflict edge*, the deepest sag the data center can both ride through and hold the required floor. We compute the frontier with inspiration from two established directions of work. Parametric STL treats values inside a temporal-logic spec as parameters [4], often used in specification mining to derive the tightest spec a fixed model satisfies [19, 24]; we likewise treat the active-power floor and the voltage sag depth as parameters of the STL spec, but the inner oracle at each parameter value is whether a controller (2) exists, instead of whether a fixed model satisfies the formula. Multiparametric programming characterizes how the solution of an optimization varies with its parameters [8], used in explicit model predictive control to precompute the control law as a function of state; we likewise sweep these parameters, but keep only the boundary of the feasible region as the conflict frontier, instead of the control law across it.

Diagnosis. To attribute an infeasibility to specific clauses, we recompute (2) over subsets of the guarantee clauses and isolate the minimal infeasible subset, which is the set of clauses for which no controller exists but dropping any single clause makes the rest feasible. This is the Irreducible Infeasible Subsystem (IIS) of the program [11]. IIS extraction usually explains an infeasible program to the author; in our case, we leverage IIS to identify the set of grid code clauses in conflict, turning the infeasibility reported by the solver into a diagnosis. With three clauses, as in the ERCOT example, every subset can be tested with ease. If a grid code presents more guarantee clauses, we can rely on the solver to extract an irreducible infeasible subset on its own [18]. A clause that is infeasible on its own cannot be satisfied with any controller. Clauses that are feasible individually but infeasible together are in conflict, such as the connection and active power floor clauses competing for the apparent power rating S.

Provisioning. Provisioning closes the gap between the requirement and the frontier by raising the frontier or lowering the requirement through three knobs. 1) The UPS/BESS power-conversion-system (PCS) apparent-power rating S raises the frontier directly: a larger apparent power rating allows more P and Q simultaneously, with more reactive power headroom to maintain internal voltage and more active power to maintain the active power floor. 2) As part of the plant model, increasing the interconnection reactance of lines and transformers raises the converter's voltage authority: a larger reactance gives more internal-voltage change per unit of reactive power injection, so lower Q holds the internal voltage at a given sag depth and more of the rating S is left for P. 3) Adjusting the workload power profile acts on the requirement: lowering the pre-fault draw p_{pre} lowers the active power floor requirement during the fault, lowering the grid code requirement curve against the conflict frontier. Each of the three knobs is monotone, allowing us to apply binary search to derive the value at which the query first becomes feasible, i.e., the smallest change that restores compliance.

4 Results

The evaluation answers three questions: (a) on a live grid, does the synthesized controller comply with the grid code where alternative control schemes fail (§4.1); (b) when the synthesis query is infeasible, can SoLVRT diagnose which clauses conflict or whether any individual clause is unsatisfiable (§4.2); and (c) what is the smallest hardware or workload change that restores compliance (§4.3)?

Setup. We evaluate on a 200 MW data center instantiated from the Vulcan facility data [39]: eight compute nodes, each modeled as a 20 MW load supplied through an upstream UPS with a 25 MVA grid-facing converter; two 20 MVA battery (BESS) units sit at the facility's distribution bus; and a 40 MW cooling plant brings the rated facility draw to 200 MW. The compute draw follows the measured power of a synchronous AI training job [12], normalized to peak and scaled to the facility's compute capacity.

Live grid simulator. The controller of §3.3 is synthesized against the worst-case trace of the grid code envelope; §4.1 runs it closed-loop on a grid simulator, which changes its input in two ways. First, the measured voltage is one realized transient inside the assumed envelope, produced by the grid model, not the worst-case bound of (7). Second, the loop is closed: the data center's own P and Q enter the grid's power-flow solution and change the grid voltage the data center measures next. We connect the data center to ANDES [13] on the 140-bus NPCC system on bus 128. The disturbance is a three-phase fault near the interconnection, applied at t=1 s and held for its band's ride-through duration before clearing (2 s for a sag in $[0.8, 0.9)$, 0.5 s for sags in $[0.5, 0.8)$). The closer the fault, the deeper the interconnection voltage sags; we vary the fault impedance on the transmission grid to simulate faults at different electrical distances, producing voltages of different depths inside the envelope the code assumes. Every 0.02 s the controller reads the interconnection voltage, re-solves with the measurement held constant over its horizon, converts the first action to aggregate

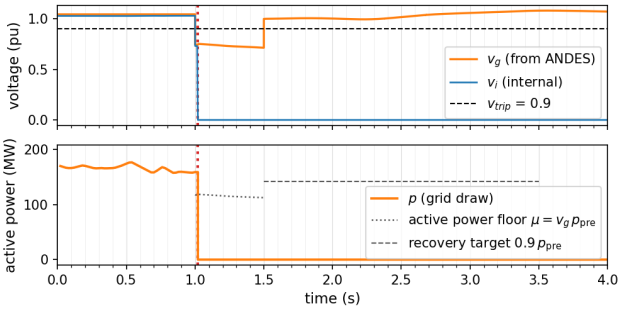


Figure 5: No-controller baseline. The fault at t=1 s trips the data center; both the internal voltage and the active power draw drop to zero, violating all three clauses.

P and Q, and writes them to the load; ANDES takes the P and Q and advances, and the next interconnection voltage measurement closes the loop. In every run, the facility's undervoltage protection disconnects the data center if the internal voltage v_i remains below $v_{\text{trip}} = 0.9$ for 0.02 s, and a disconnected facility draws zero power.

Experiment design. The subsections map to the three questions and isolate one variable at a time. To answer (a), §4.1 varies the controller on a provisioned variant of the facility with converters rated at S=40 MVA per node, large enough that a compliant controller exists at every voltage sag depth the grid code covers that requires an active power floor: no controller and a penalty-based controller from prior work [44] serve as baselines against the controller synthesized from (2), each run with the live grid simulator on the same faults. To answer (b) and (c), §4.2 and §4.3 return to the as-built 25 MVA rating, where the synthesis query is infeasible and the diagnosis and provisioning of §3.4 take over.

4.1 Ride-through Effectiveness on a Live Grid

No controller (Figure 5). With the UPS and BESS taking no control action to inject active or reactive power, the internal voltage follows the grid voltage. The fault sags v_g to 0.71 and v_i follows; after the 0.02 s pickup the protection disconnects the facility. All three clauses fail together: v_i drops below v_{trip} and the data center disconnects (connection); as a result, the 160 MW draw drops to zero, below the ≈110 MW floor (floor), and it never returns to $0.9\, p_{\text{pre}}$ within the 2 s deadline (recovery). The grid recovers after the fault is cleared and v_g settles above its pre-fault value: the disconnection becomes a disturbance. At one facility the post-fault voltage overshoot is modest, but it becomes a systemic event when a cluster of facilities trips at the same time, as observed in the Virginia incident in 2024 [32], the behavior the code is written to prevent.

Penalty controller (Figure 6). Penalty-based voltage control is the standard alternative, which can hold the internal voltage to avoid tripping, but compliance depends on a hand-tuned weight and carries no guarantee. We implement a data-center voltage controller from prior work [44]: at each step, the control law chooses the converter's P and Q injections that pull the internal voltage toward nominal, with a weight c penalizing deviation from normal operation. It's expressed as $\min_u \|v_i - \mathbf{1}\|^2 + c\,\|u - u_{\text{nom}}\|^2$ over the same feasible set K, where u_{nom} is normal operation: the converters simply import compute draw and provide no reactive support. This

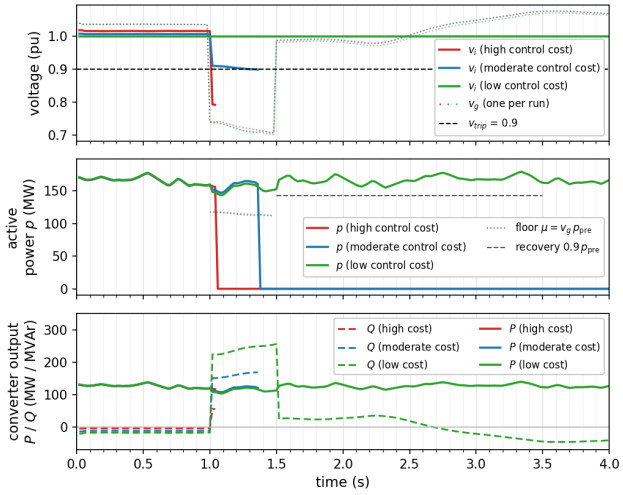


Figure 6: Penalty-based controller under three penalty weights on the same fault. The grid voltage v_g differs across runs because the loop is closed. Controller with high cost trips data center at the fault onset; one with moderate cost holds v_i near 0.90 and trips mid-fault at t=1.34 s; one with low cost rides through.

is the objective of the controller that Xie et al. used [44], written in our notation and with their per-device effort weights collapsed to the single scalar c and their box bounds replaced by K, so the penalty controller commands the same hardware as the synthesized one. In such a penalty-based controller, no grid code clause of φ appears in the objective. We run three weights on the same fault as the no-controller run. At c=5 (high control cost) the controller is reluctant to act, with v_i reaching only 0.79 and the data center tripping at the fault onset. At c=0.001 (low control cost) it holds $v_i \approx 1$ and the active power draw stays above the floor as a result, which is grid code compliant. At c=0.8 (medium control cost), the controller holds $v_i \approx 0.90$ with almost no margin through the first 0.3 s of the fault, then v_i slips under the threshold and the data center trips at t=1.34 s; the active power draw drops to zero, and the floor and recovery clauses fail with it. Compliance thus depends entirely on the hand-picked weight, and the formulation gives no prior indication of the compliant range: a weight that survives the fault onset can fail partway through, and the floor and recovery are met only incidentally. In our case, the synthesis query (2) decides compliance offline before the fault instead of during it: it either returns a compliant controller by construction, or reports infeasibility at design time with no compliant controller.

Synthesized controller (Figure 7). The synthesized controller rides through three voltage sag depths inside the assumed voltage envelope and complies with the grid code. During each fault, the controller holds v_i slightly above v_{trip} (a 0.005 pu robustness margin), injecting more reactive power the deeper the sag: roughly 70, 170, and 300 MVAr at depths 0.83, 0.70, and 0.54. The converter active power simply tracks the compute demand and meets the floor with no curtailment, because at this rating the reactive support and the full compute load fit inside the apparent-power headroom together, so compliance with active power floor as well as recovery

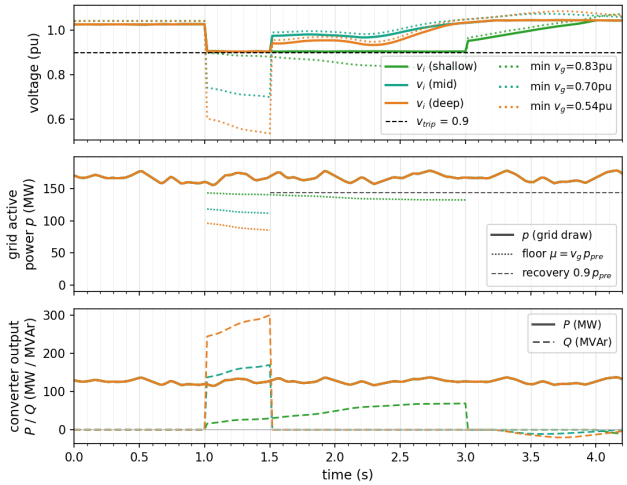


Figure 7: Synthesized controller on three sag depths. Thin dotted curves in the top panel are each run's grid voltage v_g; the shallow sag is held for its band's 2 s duration and the mid and low sags for 0.5 s, all inside the ERCOT envelope. In every run the controller holds v_i just above v_{trip} while the draw p tracks the workload above its floor μ.

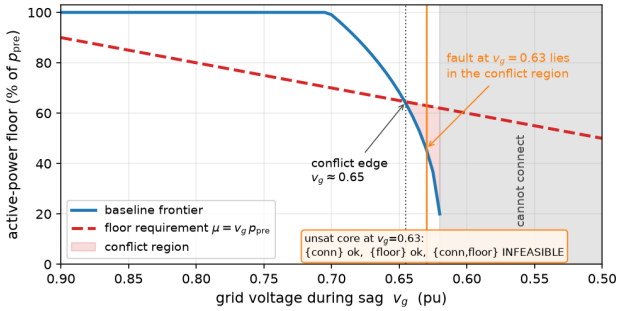


Figure 8: Conflict frontier of the as-built facility, swept at 81 depths (0.005 pu apart). The frontier is capped at 100% of p_{pre}, since a floor above the pre-fault draw is meaningless. The orange marker at $v_g = 0.63$ is the depth examined in Figure 10, with its clause-subset diagnosis in the box.

costs no active power control action. The guarantee is conditional on exactly this headroom, where the facility's 40 MVA converters sit above the minimum that §4.3 derives; on the as-built 25 MVA rating, the P and Q no longer fit together and the query returns infeasible and no controller exists. The next subsections turn that infeasibility into a diagnosis and a provisioning decision.

4.2 Infeasibility Diagnosis

This subsection and the next fix the controller and return to the baseline data center with the as-built 25 MVA rating in the Vulcan system [39]. The question is design-time feasibility, so no grid simulator is involved. The synthesis query of §3.3 takes three inputs. The specification is the ERCOT code of §3.2. The plant model is the Vulcan facility at its as-built rating, modeled with LinDistFlow. The grid-voltage disturbance is the worst-case envelope of (7), restricted to the depths where the active power floor applies ($v_g \geq 0.5$); below 0.5 the code releases the active power floor and permits momentary cessation, the facility briefly stopping its grid power draw and restoring it once the voltage returns, so no converter action is required during the fault. The workload disturbance is the measured trace with the grid fault aligned to the trace peak. A fault can land anywhere on the trace, and p_{pre}, the active power draw at the fault instant, sets the floor $\mu = v_g\, p_{\text{pre}}$, so this alignment is the worst case, $p_{\text{pre}} = 200$ MW. The synthesis query returns infeasible for this combination of grid code spec, plant model, and disturbance, meaning no controller satisfies φ.

The conflict frontier (Figure 8). The synthesis against the as-built facility fails in two ways, requiring different fixes. At moderate sag depths, the connection and active power floor clauses conflict: each clause is satisfiable alone, but their conjunction is not (the minimal infeasible subset of §3.4). At deeper sags, connection is infeasible alone. Recovery is never among the failing clauses. We locate the two modes with the conflict frontier of §3.4, finding at each depth between 0.90pu and 0.50pu the largest floor the data center can sustain while staying connected, and re-solving the query over clause subsets at each depth. The required floor crosses the frontier at $v_g \approx 0.65$pu, the conflict edge. Between 0.65pu and 0.62pu the data center can either stay connected or maintain the active power floor, but not both. Below 0.62pu, holding internal voltage to stay connected alone is infeasible, at what we call the connection wall. At the wall the frontier ends at 40 MW (20% of p_{pre}) rather than zero, where the converters are fully spent on reactive support, and the residual is the cooling draw, which is served without a converter and sits outside the control knobs we model. Inside the conflict region, raising the capability or lowering the requirement both restore feasibility; below the wall, only raising the capability helps.

The conflict on the time axis. At $v_g = 0.63$, satisfying the connection clause takes 234 MVAr of reactive power, which nearly fills the converter headroom and leaves active power for a draw of only 92 MW, 34 MW below the 126 MW floor. Appendix B plots this trajectory in the time domain.

4.3 Provisioning

For a provisioning team, this subsection answers the question of which hardware to specify so that the facility is capable of complying with the LVRT grid code. Each knob is monotone, so a binary search on the synthesis query returns the smallest value at which it turns feasible, a design requirement derived directly from the grid code and available before the facility is built or upgraded.

Provisioning, hardware (Figure 9a). Provisioning closes the gap between the requirement and the capability. §3.4 exposes two hardware knobs and they help raise the frontier: the UPS converter rating S and the interconnection reactance. A larger S widens the eight per-node apparent power rating headroom with more Q for connection and more P for the floor at the same time. A larger reactance moves more internal voltage per unit of Q, so less of each headroom holds v_i and more remains for P. Both are monotone and bisection returns the minimal change, S from 25 to 38 MVA per node (+50%) or reactance scaled by 1.49 in the model. Both hardware knobs lift the frontier above the requirement at every depth down to $v_g = 0.50$, the bottom of the active power floor band.

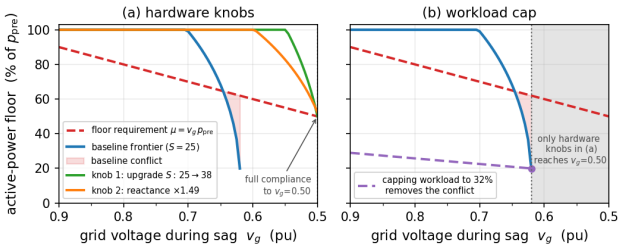


Figure 9: Provisioning. (a) Hardware: each knob is raised to the minimal value that restores compliance, the converter rating S from 25 to 38 MVA or the interconnection reactance scaled by 1.49; either lifts the frontier above the code at every depth down to $v_g = 0.50$. (b) Workload: at the minimal cap of 32% of rated draw, the capped requirement falls below the frontier at every depth where the facility can connect, removing the conflict that is above the the connection wall.

Both knobs also push the connection wall below 0.50. The BESS apparent power rating is a separate yet weaker knob, so we don't demonstrate in Figure 9. Because BESS reactive power reaches the internal buses only through the facility trunk, its reactive power injection barely helps maintain internal voltage to avoid tripping.

Provisioning, workload (Figure 9b). The workload enters this analysis only through the pre-fault draw. During a fault the UPS battery carries the compute, which costs little energy over events of a few seconds, and the grid-side draw becomes a control that the synthesis sets freely, up to the converter rating. What the workload does during the fault therefore never appears in the fault-time constraints, but only as p_{pre}, the active power draw at the fault instant, which sets the requirement $\mu = v_g\, p_{\text{pre}}$. The software knob caps the pre-fault workload. The cap lowers p_{pre} and the requirement with it and leaves the frontier where it is. In our model, capping the draw at 32% of rated ($p_{\text{pre}} = 64$ MW) brings the requirement below the capability at every depth the data center can connect, with no hardware change, removing the $\{\varphi_{\text{conn}}, \varphi_{\text{flr}}\}$ conflict entirely. However, the connection wall does not move with software changes alone, as it stays constrained by the maximum reactive power the UPS hardware can inject; no software capping connects the data center below 0.62 and compliance down to 0.50 stays infeasible.

The contrast summarizes the provisioning section. Hardware raises data center capability by moving both the frontier and the connection wall. A workload cap only lowers the grid code floor requirement and is still bounded by the hardware constraints. Hardware provisioning is the more fundamental fix, as workload capping alone is only a stop-gap for sags above the connection wall.

5 Discussion

Grid code ambiguity. Formalizing ERCOT's grid code exposed two underspecified terms. First, the code scales its active power floor and recovery requirements to the facility's "pre-disturbance level" but does not define how that value is read. Over a fluctuating AI workload, different methods of deriving the "pre-disturbance level" could give drastically different values. We take the instantaneous active power draw (§3.1) for this work, instead of taking an average over some window. Second, the recovery clause requires the facility to "return to 90% of its pre-disturbance level within two seconds," but presumably, a facility that touches the level at the deadline and drops off immediately still violates the code. We add the hold window T_{hold} of (6) so that recovery means genuine restored consumption rather than a momentary spike. These nuances emerged only when translating the natural-language grid code into a formal specification. This highlights the value of a formal approach, as it forces ambiguous requirements to surface and be made explicit before they are enforced.

Modeling state-of-charge. We do not model energy storage state-of-charge (SoC). Batteries in UPS and BESS may charge or discharge during ride-through. On the discharging side, riding-through a voltage event spends a small fraction of the stored energy, since a VRT event lasts only a few seconds, while the UPS's capacity is provisioned to carry the full compute load for tens of seconds and the BESS for minutes. Charging requires batteries not held at full charge, and the headroom needed is likewise a small fraction of their provisioned capacity. Thus, we do not model the SoC, and assume the UPS and BESS have both enough stored energy for discharging and enough headroom for charging during a VRT event.

Beyond one code and one facility. Similar to LVRT grid codes, other grid codes regulate data center behavior for other types of grid disturbances (e.g., frequency ride-through). ERCOT writes frequency ride-through in the same document as voltage ride-through with the same structure: an assumption envelope on grid frequency, a power draw requirement dependent on the pre-disturbance level, and a power draw recovery clause. SoLVRT has the potential to solve frequency ride-through with minimal changes: our specification language can express these clauses with the same shape, just with frequency declared as the disturbance signal instead; however, the plant model and control knobs differ, and we leave the adaptation to future work. On the data center side, while SoLVRT certifies individual data centers' compliance with the LVRT grid code, the aggregated behavior of clustered data centers and their impact on the grid remains an open grid-side question.

6 Conclusion

Data centers have grown into loads large enough that their protective disconnection threatens the grid during voltage disturbances, and emerging VRT grid codes now require them to ride through such voltage events. This paper solves data center compliance with VRT grid codes using a controller synthesis approach. SoLVRT expresses a grid code as an assume-guarantee specification in Signal Temporal Logic, synthesizes a controller that complies by construction whenever one exists, and turns infeasibility into a conflict frontier, a diagnosis of the conflicting clauses, and a provisioning recommendation for the smallest hardware or workload change that restores compliance. On a model of a 200 MW facility in a closed loop with a transmission grid simulator, the synthesized controller rides through faults that trip the uncontrolled facility, and the diagnosis traces the as-built rating's infeasibility to converter capacity, for which a hardware upgrade restores feasibility. As grid operators write ride-through rules for large loads and data center operators adapt to comply, SoLVRT gives both sides a machine-checkable interface between grid requirements and data center capabilities.

Ethics Statement

This work does not raise ethical issues, as it relies on published data and involves no human subjects or personal data.

Acknowledgments

We thank the anonymous reviewers for their valuable feedback. This work was supported in part by NSF awards CNS 2535540, 2406598, 2420309, and 2345339, a Cisco gift, and the UM Bold Challenges award.

References

[1] Yousef Abudyak, Mohsen Alizadeh, and Wei Sun. 2026. Mitigating Data Centers Load Risks and Enabling Grid Support Functions through Grid-Forming Control. arXiv:2602.01013.

[2] Fatih Acun, Can Hankendi, Ethan Levine, Hudson Reynolds, Joshua Bardwick, and Ayse K. Coskun. 2026. Investigating Power Consumption Flexibility of AI Data Centers for Demand Response Participation. In *Proceedings of the 17th ACM International Conference on Future and Sustainable Energy Systems*. ACM, Banff , Alberta , Canada, 344–348. doi:10.1145/3744255.3798112

[3] Baris Aksanli, Alper Sinan Akyurek, and Tajana Rosing. 2015. Minimizing the effects of data centers on microgrid stability. In *2015 Sixth International Green and Sustainable Computing Conference (IGSC)*. IEEE, Las Vegas, NV, USA, 1–9. doi:10.1109/IGCC.2015.7393720

[4] Eugene Asarin, Alexandre Donzé, Oded Maler, and Dejan Nickovic. 2012. Parametric Identification of Temporal Properties. In *Runtime Verification*, Sarfraz Khurshid and Koushik Sen (Eds.). Springer, Berlin, Heidelberg, 147–160. doi:10.1007/978-3-642-29860-8_12

[5] Mesut E. Baran and Felix F. Wu. 1989. Optimal capacitor placement on radial distribution systems. *IEEE Transactions on Power Delivery* 4, 1 (1989), 725–734. doi:10.1109/61.19265

[6] Ezio Bartocci, Jyotirmoy Deshmukh, Alexandre Donzé, Georgios Fainekos, Oded Maler, Dejan Ničković, and Sriram Sankaranarayanan. 2018. Specification-Based Monitoring of Cyber-Physical Systems: A Survey on Theory, Tools and Applications. In *Lectures on Runtime Verification: Introductory and Advanced Topics*, Ezio Bartocci and Yliès Falcone (Eds.). Springer International Publishing, Cham, 135–175. doi:10.1007/978-3-319-75632-5_5

[7] Calin Belta and Sadra Sadraddini. 2019. Formal Methods for Control Synthesis: An Optimization Perspective. *Annual Review of Control, Robotics, and Autonomous Systems* 2 (2019), 115–140.

[8] Alberto Bemporad, Manfred Morari, Vivek Dua, and Efstratios N. Pistikopoulos. 2002. The explicit linear quadratic regulator for constrained systems. *Automatica* 38, 1 (Jan. 2002), 3–20. doi:10.1016/S0005-1098(01)00174-1

[9] Albert Benveniste, Benoît Caillaud, Dejan Nickovic, Roberto Passerone, Jean-Baptiste Raclet, Philipp Reinkemeier, Alberto Sangiovanni-Vincentelli, Werner Damm, Thomas A. Henzinger, and Kim G. Larsen. 2018. Contracts for System Design. *Foundations and Trends in Electronic Design Automation* 12, 2–3 (2018), 124–400.

[10] Yuxiao Chen, James Anderson, Karan Kalsi, Aaron D. Ames, and Steven H. Low. 2019. Compositional Set Invariance in Network Systems with Assume-Guarantee Contracts. In *American Control Conference (ACC)*. 1027–1034.

[11] John W Chinneck and Erik W Dravnieks. 1991. Locating minimal infeasible constraint sets in linear programs. *ORSA Journal on Computing* 3, 2 (1991), 157–168.

[12] Esha Choukse, Brijesh Warrier, Scot Heath, Luz Belmont, April Zhao, Hassan Ali Khan, Brian Harry, Matthew Kappel, Russell J. Hewett, Kushal Datta, Yu Pei, Caroline Lichtenberger, John Siegler, David Lukofsky, Zaid Kahn, Gurpreet Sahota, Andy Sullivan, Charles Frederick, Hien Thai, Rebecca Naughton, Daniel Jurnove, Justin Harp, Reid Carper, Nithish Mahalingam, Srini Varkala, Alok Gautam Kumbhare, Satyajit Desai, Venkatesh Ramamurthy, Praneeth Gottumukkala, Girish Bhatia, Kelsey Wildstone, Laurentiu Olariu, Ileana Incorvaia, Alex Wetmore, Prabhat Ram, Melur Raghuraman, Mohammed Ayna, Mike Kendrick, Ricardo Bianchini, Aaron Hurst, Reza Zamani, Xin Li, Michael Petrov, Gene Oden, Rory Carmichael, Tom Li, Apoorv Gupta, Pratikkumar Patel, Nilesh Dattani, Lawrence Marwong, Rob Nertney, Hirofumi Kobayashi, Jeff Liott, Miro Enev, Divya Ramakrishnan, Ian Buck, and Jonah Alben. 2025. Power Stabilization for AI Training Datacenters. arXiv:2508.14318 [cs.AR] https://arxiv.org/abs/2508.14318

[13] Hantao Cui, Fangxing Li, and Kevin Tomsovic. 2021. Hybrid Symbolic-Numeric Framework for Power System Modeling and Analysis. *IEEE Transactions on Power Systems* 36, 2 (2021), 1373–1384. doi:10.1109/TPWRS.2020.3017019

[14] Alexandre Donzé and Vasumathi Raman. 2015. BluSTL: Controller Synthesis from Signal Temporal Logic Specifications. In *1st and 2nd International Workshop on Applied Verification for Continuous and Hybrid Systems (ARCH)*. 160–168.

[15] Energinet. 2024. *Technical Regulation 3.4.3 Requirements for Transmission-Connected Demand Facilities, Revision 1.* Technical Report 3.4.3. Energinet. https://en.energinet.dk/media/bkpf4hef/technical-regulation-343-requirements-for-transmission-connected-demand-facilities-revision-1.pdf

[16] ERCOT. 2026. *Board Priority - Large Computational Load Ride-Through Requirements.* Nodal Operating Guide Revision Request 282NOGRR-30. Electric Reliability Council of Texas. https://www.ercot.com/files/docs/2026/05/11/282NOGRR-30-ERCOT-Comments-051126.docx

[17] Georgios E. Fainekos and George J. Pappas. 2009. Robustness of Temporal Logic Specifications for Continuous-Time Signals. *Theoretical Computer Science* 410, 42 (2009), 4262–4291.

[18] Gurobi Optimization, LLC. 2026. `Model.computeIIS`: Python API Reference. *Gurobi Optimizer Reference Manual*. Gurobi Optimizer 13.0 documentation. https://docs.gurobi.com/projects/optimizer/en/current/reference/python/model.html#Model.computeIIS

[19] Bardh Hoxha, Adel Dokhanchi, and Georgios Fainekos. 2018. Mining parametric temporal logic properties in model-based design for cyber-physical systems. *International Journal on Software Tools for Technology Transfer* 20, 1 (Feb. 2018), 79–93. doi:10.1007/s10009-017-0447-4

[20] IEEE. 2022. IEEE Standard 2800-2022: Interconnection and Interoperability of Inverter-Based Resources Interconnecting with Associated Transmission Electric Power Systems.

[21] Inigo Incer. 2022. *The Algebra of Contracts*. Ph. D. Dissertation. EECS Department, University of California, Berkeley. http://www2.eecs.berkeley.edu/Pubs/TechRpts/2022/EECS-2022-99.html

[22] Inigo Incer, Albert Benveniste, and Alberto Sangiovanni-Vincentelli. 2023. Some Algebraic Aspects of Assume-Guarantee Reasoning. *arXiv 2309.08875* (2023). arXiv:2309.08875 [cs.LO] https://arxiv.org/abs/2309.08875

[23] Inigo Incer, Noel Csomay-Shanklin, Aaron D. Ames, and Richard M. Murray. 2024. Layered Control Systems Operating on Multiple Clocks. *IEEE Control Systems Letters* 8 (2024), 1211–1216. doi:10.1109/LCSYS.2024.3410150

[24] Xiaoqing Jin, Alexandre Donzé, Jyotirmoy V. Deshmukh, and Sanjit A. Seshia. 2015. Mining Requirements From Closed-Loop Control Models. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 34, 11 (Nov. 2015), 1704–1717. doi:10.1109/TCAD.2015.2421907

[25] Hadas Kress-Gazit, Morteza Lahijanian, and Vasumathi Raman. 2018. Synthesis for Robots: Guarantees and Feedback for Robot Behavior. *Annual Review of Control, Robotics, and Autonomous Systems* 1 (2018), 211–236.

[26] Soumya Kundu, Kaustav Chatterjee, Rakib R. Hossain, Sai Pushpak Nandanoori, and Veronica Adetola. 2025. Managing Risks from Large Digital Loads Using Coordinated Grid-Forming Storage Network. arXiv:2508.11080.

[27] Zhirui Liang, Jae-Won Chung, Mosharaf Chowdhury, Jiasi Chen, and Vladimir Dvorkin. 2026. GPU-to-Grid: Voltage Regulation via GPU Utilization Control. arXiv:2602.05116.

[28] Zhenhua Liu, Adam Wierman, Yuan Chen, Benjamin Razon, and Niangjun Chen. 2013. Data center demand response: avoiding the coincident peak via workload shifting and local generation. In *Proceedings of the ACM SIGMETRICS/international conference on Measurement and modeling of computer systems*. ACM, Pittsburgh PA USA, 341–342. doi:10.1145/2465529.2465740

[29] Oded Maler and Dejan Nickovic. 2004. Monitoring Temporal Properties of Continuous Signals. In *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems (FORMATS/FTRTFT)*. 152–166.

[30] Sina Mohammadi, Wayne Wang, Marcus Chen I Wada, Rouzbeh Haghighi, Ali Hassan, Hualong Liu, Archit Bhatnagar, Ang Chen, and Wencong Su. 2026. Grid Integration of AI Data Centers: A Critical Review of Energy Storage Solutions. arXiv:2603.00415 [eess.SY] https://arxiv.org/abs/2603.00415

[31] Amir Norouzi and Michael Morel. 2026. Voltage Ride-Through in Large Loads: A Dual PQ Approach. arXiv:2605.00769 [eess.SY] https://arxiv.org/abs/2605.00769

[32] North American Electric Reliability Corporation. 2025. *Incident Review: Considering Simultaneous Voltage-Sensitive Load Reductions.* Technical Report. North American Electric Reliability Corporation. https://www.nerc.com/globalassets/our-work/reports/event-reports/incident_review_large_load_loss.pdf

[33] Pierluigi Nuzzo, Huan Xu, Necmiye Ozay, John B. Finn, Alberto L. Sangiovanni-Vincentelli, Richard M. Murray, Alexandre Donzé, and Sanjit A. Seshia. 2014. A Contract-Based Methodology for Aircraft Electric Power System Design. *IEEE Access* 2 (2014), 1–25.

[34] Vasumathi Raman, Alexandre Donzé, Mehdi Maasoumy, Richard M. Murray, Alberto Sangiovanni-Vincentelli, and Sanjit A. Seshia. 2014. Model Predictive Control with Signal Temporal Logic Specifications. In *53rd IEEE Conference on Decision and Control (CDC)*. 81–87. doi:10.1109/CDC.2014.7039363

[35] Vasumathi Raman, Alexandre Donzé, Dorsa Sadigh, Richard M. Murray, and Sanjit A. Seshia. 2015. Reactive Synthesis from Signal Temporal Logic Specifications. In *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control (HSCC)*. 239–248. doi:10.1145/2728606.2728628

[36] Nicolas Rouquette, Inigo Incer, and Alessandro Pinto. 2023. Early Design Exploration of Space System Scenarios Using Assume-Guarantee Contracts. In *2023 IEEE International Conference on Space Mission Challenges for Information Technology (SMC-IT)*. IEEE, Pasadena, CA, USA, 15–24. doi:10.1109/SMC-

IT56444.2023.00011
[37] Mohamed Shamseldein. 2025. From Liability to Asset: A Three-Mode Grid-Forming Control Framework for Centralized Data Center UPS Systems. arXiv:2512.16497.
[38] Marina Tsili and Stavros Papathanassiou. 2009. A Review of Grid Code Technical Requirements for Wind Farms. *IET Renewable Power Generation* 3, 3 (2009), 308–332.
[39] Deepthi Vaidhynathan, Kumaraguru Prabakar, Gregory Martin, Anand Ramesh, Ben Wheeler, Christopher Coco, Jimmy Clidaras, Matthieu Monsch, Sangsun Kim, Saurav Talukdar, and Shilpa Marti. 2025. Vulcan Test Platform: Demonstrating the Data Center as a Flexible Grid Asset. *Renewable Energy* (2025).
[40] Jesus D. Vasquez-Plaza, Yonghao Gui, Jin Dong, Jamie Lian, and Yilu Liu. 2026. Keeping Data Centers Online in Weak Grids: PLL-Free VM-DPC With Adaptive Reactive-Power Support for Centralized UPS Systems. arXiv:2608.06521 [eess.SY] https://arxiv.org/abs/2608.06521
[41] Di Wang, Sriram Govindan, Anand Sivasubramaniam, Aman Kansal, Jie Liu, and Badriddine Khessib. 2014. Underprovisioning backup power infrastructure for datacenters. In *Proceedings of the 19th international conference on Architectural support for programming languages and operating systems*. ACM, Salt Lake City Utah USA, 177–192. doi:10.1145/2541940.2541966
[42] Lizhi Wang, Fei Feng, Ella Chou, and Yashen Lin. 2026. Resilient Energy-Based Control for DC Data Centers under Grid and Load Disturbances. arXiv:2605.21691.
[43] Adam Wierman, Zhenhua Liu, Iris Liu, and Hamed Mohsenian-Rad. 2014. Opportunities and challenges for data center demand response. In *International Green Computing Conference*. 1–10. doi:10.1109/IGCC.2014.7039172
[44] Yiheng Xie, Wenqi Cui, and Adam Wierman. 2026. Enhancing Data Center Low-Voltage Ride-Through. In *Proceedings of the 17th ACM International Conference on Future and Sustainable Energy Systems (e-Energy)*. arXiv:2510.03867.
[45] Zhe Xu, A. Agung Julius, and Joe H. Chow. 2017. Optimal Energy Storage Control for Frequency Regulation under Temporal Logic Specifications. In *American Control Conference (ACC)*. 1874–1879.
[46] Zhe Xu, A. Agung Julius, and Joe H. Chow. 2019. Energy Storage Controller Synthesis for Power Systems with Temporal Logic Specifications. *IEEE Systems Journal* 13, 1 (2019), 748–759.
[47] Yichen Zhang, Mohammed Olama, Alexander Melin, Yaosuo Xue, Seddik Djouadi, and Kevin Tomsovic. 2019. Synthesizing Distributed Energy Resources in Microgrids with Temporal Logic Specifications. In *IEEE Innovative Smart Grid Technologies (ISGT)*. 1–5.
[48] Hao Zhu and Hao Jan Liu. 2016. Fast Local Voltage Control Under Limited Reactive Power: Optimality and Stability Analysis. *IEEE Transactions on Power Systems* 31, 5 (2016), 3794–3803. doi:10.1109/TPWRS.2015.2504419

Appendices are supporting material that has not been peer-reviewed.

Algorithm 1 Encoding a specification to a synthesis query.

function ENCODE($spec$, plant, workload w_{comp})
  if $spec$.assume declares envelope $\{(v_k, d_k)\}_{k=1}^{n}$ **then**
    $v'_g(t) \leftarrow \min\{\, v_k : d_k \geq t \,\}$ ▷ Eq. (7)
  else
    $v'_g \leftarrow spec$.assume ▷ already a piecewise lower bound
  $t_f \leftarrow \min\{\, t : v'_g(t) < 0.9 \,\}$
  $t_r \leftarrow \min\{\, t > t_f : v'_g(t) \geq 0.9 \,\}$ ▷ event times from v'_g
  $w \leftarrow (v'_g,\ w_{comp})$ ▷ disturbance inputs
  $(v_i, p) \leftarrow \text{plant}(u, w)$ ▷ the plant model mapping $y = g(u, w)$

  $vals \leftarrow \emptyset$
  for all binding $b \in spec$.var **do** ▷ instantiate binding
    $vals \leftarrow vals \cup \{\, b[\, t_f, t_r, w \,]\, \}$ ▷ e.g. $p_{pre} = w_{comp}(t_f)$
  $\varphi \leftarrow \emptyset$
  for all clause $c \in spec$.guarantee **do** ▷ instantiate clause
    $\varphi \leftarrow \varphi \cup \{\, c[\, t_f, t_r, vals \,]\, \}$

  $K \leftarrow \{\, u : P^2 + Q^2 \leq S^2 \,\}$ ▷ apparent-power cap, S from plant
  return SYNTHESIZE($(v_i,\ p)$, φ, K, w) ▷ solve (2)

A The Encoding Procedure

Algorithm 1 lists the full encoding procedure of §3.3. It constructs the worst-case trace from the specification's assume block, resolves the event times, instantiates the declared variables and the guarantee clauses, binds the plant outputs, and calls the synthesis query (2).

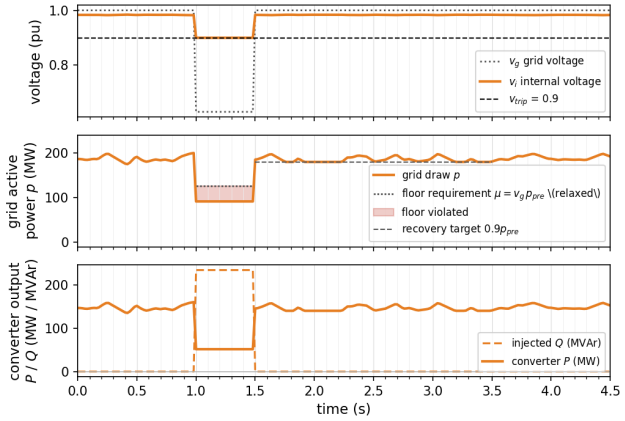


Figure 10: The conflict at $v_g = 0.63$. Only the connection clause is enforced for demonstration; with best effort on the active power clause, the converter headroom is not enough for active power to satisfy the floor clause, causing a violation.

B The Conflict on the Time Axis

Figure 10 shows what the conflict of §4.2 means in the time domain: even the largest active power draw the facility can sustain while staying connected falls short of the floor the spec requires. An infeasible query returns no controller, so there is no control action trajectory to see the violation. We therefore plot a relaxation instead, by keeping enforcement on the connection clause and making the best effort toward the active power floor clause. Staying connected takes 234 MVAr of reactive power Q, which nearly fills the converter headroom S, and the remaining active power P supports an active power draw of only 92 MW, 34 MW below the 126 MW floor requirement. In other words, the connection clause is only satisfiable by giving up the active power floor clause inside the conflict region.