Bringing Distributed Energy Storage to Market
Christoph Goebel, Member, IEEE, and Hans-Arno Jacobsen, Member, IEEE

Abstract—Spatially distributed energy storage devices can provide additional flexibility to system operators, which is needed to transition from primarily fossil fuel based electricity generation to variable renewable generation. Aggregators in charge of controlling distributed energy storage can take advantage of existing economic incentives for more flexibility. However, controlling large numbers of energy storage devices with individual constraints in accordance with the strict rules of existing energy and reserve markets is challenging. The purpose of this article is to investigate the design and performance of a system that enables aggregators to bring large numbers of dedicated and fully controllable energy storage devices to multiple markets concurrently. In particular, we propose algorithms and heuristic optimization methods that allow aggregators to control such energy resources in accordance with arbitrary market rules and participation strategies. Our evaluation is based on a realistic dual market (reserve and intra-day energy) use case. We find that effective market-conform control of large numbers of energy storage devices using the proposed algorithms is feasible, even on short time scales. Furthermore, our results also indicate that the scalability of the proposed system design can be further improved via parallelization without limiting the reserve/energy brought to market.

Index Terms—Energy management, energy storage, electricity markets, optimization methods, scheduling algorithms.

NOMENCLATURE

- $i$ Resource ID.
- $k$ Bid ID.
- $n$ Number of storage devices.
- $t$ Variable to denote time.
- $d$ Variable to denote duration; default unit: min.
- $l$ Variable to denote energy storage level; unit: fraction.
- $\Delta$ Variable to denote change of energy storage level; unit: fraction.

$P_i$ Maximum power of resource $i$; default unit: W.
$l_i$ Minimum energy storage level of resource $i$; default unit: W.
$t_i$ Maximum energy storage level of resource $i$; default unit: W.
$\eta$ Transformer efficiency; unit: fraction.
$e_{in}^i$ Charging rate factor of resource $i$; unit: $W \times \text{min}^{-1}$.
$e_{out}^i$ Discharging rate factor of resource $i$; unit: $W \times \text{min}^{-1}$.
$F$ Schedule fragment with parameters $i, k, l, T, p$.
$S$ Schedule with content $F_1, F_2, \ldots$
$P$ Power level with parameters $l, T, p$.
$P$ Power profile with content $P_1, P_2, \ldots$
$A_I$ Availability interval with parameters $l, T, p, \bar{p}$.
$AIS$ Set of availability intervals with content $A_{I_1}, A_{I_2}, \ldots$
$RAS$ Resource availability statement with parameters $i, \Delta, AIS$.
$RASS$ Set of resource availability statements with content $RAS_1, RAS_2, \ldots$

I. INTRODUCTION

The increasing penetration of renewable energy resources like wind and solar calls for additional flexibility, e.g., in the form of reserves [1]. Additional flexibility can be provided by directly controlling large numbers of so-called limited energy resources (LERs) in a coordinated way [2], [3]. LERs is an umbrella term that covers distributed flexible loads (e.g., electric vehicles, thermal loads in buildings, production processes), small flexible generators (e.g., small combined heat and power units), as well as small dedicated energy storage devices (e.g., chemical batteries). Recently, there has been an increasing interest in the design of system components that allow for monitoring and controlling LERs such that their potential can be fully exploited, both on the electricity distribution and transmission level. On the one hand, the potential of LERs should be leveraged locally in distribution grids, e.g., to provide voltage support or to buffer variable wind and solar energy. On the other hand, aggregators should be able to control large numbers of these resources to also provide wholesale reserve and participate in existing energy spot markets. Such smart grid systems have to be able to monitor and control devices in real time, while allowing higher-level applications for querying their current and future states based on scheduled controls. Moreover, they should provide efficient means for translating application specific commands, e.g., the command to commit a group of storage devices to stand by for reserve delivery, into feasible schedules for individual devices.

Within this research context, this article deals with the specific problem of bringing large numbers of dedicated and fully controllable energy storage devices to multiple markets, including reserve and energy markets. This task requires central control of many independent devices such that their entire
power and energy capacity can be assessed and accessed at any given time to participate in wholesale electricity markets. Our goal is to enable this type of control while satisfying all relevant market and device constraints regarding timing, power, and energy. The satisfaction of device and market constraints is crucial, no matter what the goal of the aggregator is, e.g., profit maximization, system security, or local use of renewable energy.

We make three main contributions:

- We propose effective and extensible abstractions for managing dedicated and fully controllable energy storage resources that enable concurrent participation in several markets, both energy and reserve.
- We investigate the performance of three approaches for approximating solutions of the corresponding optimization problems: a continuous optimization approach that allows for arbitrary charging or discharging rates, an integer programming approach that considers only three possible resource states (charging at maximum power, idle, and discharging at maximum power), and a greedy scheduling method.
- We evaluate the proposed system components based on a realistic dual market use case in which the aggregator controls a variable size pool of identical vanadium redox batteries with the goal of offering maximum negative reserve via a reserve market while selling the energy resulting from reserve activation via an energy spot market.

II. Model

In the following, we assume the existence of an information system infrastructure consisting of an application layer, a service layer, and a resource layer. The aggregator accesses the aggregator service via its own application. The aggregator service acts as a middleware with all necessary functionality to translate activities related to market participation into resource controls for a potentially large number of resources. Physical resources are represented by virtual resources on the service layer, i.e., there exists an instantiated software model of each physical resource on the service layer. Resource monitoring and control in the proposed system architecture can then be realized by continuous synchronization of virtual and physical resource states. When the physical state of a resource changes, e.g., when the state of charge of a battery decreases, its controller sends a corresponding message to the service layer, which in turn updates the corresponding state variables of the virtual resource. When the aggregator service updates the schedule of a resource and this update requires (physical) actuation, it sends a corresponding message to the physical resource controller. The focus of this work lies on the specification of the service level, not the design of the communication protocol that realizes state synchronization.

A. Markets and Resource Management Processes

Current market rules require that bids placed on wholesale electricity markets adhere to certain minimum sizes and size increments [4]. Bids have to be placed before certain deadlines and cannot be canceled once placed. Moreover, if reserves are activated or cleared energy market bids are physically settled, the delivered power and/or energy has to adhere to the corresponding market rules. In the case of reserves, the promised power level has to be reached within a certain time span ranging between seconds and several minutes depending on the actual reserve market [5]. Furthermore, the power consumed or delivered has to remain at the target level for as long as the system operator activates the reserve. Similarly stringent rules apply on energy markets, although in this case the focus is more on energy than power, i.e., the power delivered or consumed can vary, but the promised energy still has to be adhered to by market participants.

In the following, we formalize these constraints in a general way, i.e., our formalization describes the constraints of any type of reserve or energy market \( M \):

- If a market participant wants to buy/sell energy or sell reserve on market \( M \) during the operating interval \([t, t + d_{M}^{\text{inc}}]\), it has to place the bid before the corresponding bidding deadline \( d_{M}^{\text{bd}} \). Duration \( d_{M}^{\text{bd}} \) denotes the standardized length of operating intervals on market \( M \) and duration \( d_{M}^{\text{inc}} \) denotes the minimum time span between bidding deadline and the start of the corresponding operating interval.
- The bid size \( p_{M}^{M} \) (in MW) submitted to the market operator before time \( d_{M}^{\text{bd}} \) has to adhere to the bid size constraints of market \( M \), i.e., \( p_{M}^{M} = p_{M}^{\text{inc}} + \alpha p_{M}^{\text{inc}} \), where \( p_{M}^{\text{inc}} \) is the minimum bid size, \( p_{M}^{\text{inc}} \) is the allowed bid size increment, and \( \alpha \in \mathbb{N} \).
- The total power \( p_{M}^{\text{.bd}} \) delivered during the operating interval \([t, t + d_{M}^{\text{inc}}]\), must not deviate from the committed amount by more than a market-specific tolerance, \( \epsilon_{M} \), i.e., \( p \in \left[p_{M}^{\text{bd}} - \epsilon_{M}, p_{M}^{\text{bd}} + \epsilon_{M}\right] \).

Resource management processes describe the system activities related to the aggregator’s participation in a particular market. They can be subdivided into two types, one enabling participation in reserve markets, the other one enabling participation in energy markets. Figure 1 shows the sequence diagrams for both types. The process steps optimization \( A \) and \( B \) in Figure 1 refer to the type of optimization problem for which solutions have to be approximated by the aggregator service. Optimization goals and methods are detailed in Section III.

At any given time, the aggregator can initiate a resource management process via its application by requesting the maximum bid size it could theoretically place on a particular market in a given future time interval without violating any resource constraints or previously committed schedules, including schedule fragments pertaining to other markets. To return the maximal bid size, the service has to compute the corresponding resource availability statements (cf., Section II-B) and solve an optimization problem of type \( A \) (cf., Section III). Knowing the maximum bid size, the aggregator can then define the final market-conform bid size and trigger the commitment or dispatch of virtual resources. To translate the final bid size into desirable resource controls, the service has to solve an optimization problem of type \( B \) (cf., Section III). It then commits or dispatches virtual resources accordingly by
B. Computation of Resource Availability Statements

The aggregator service maintains a set of resource schedules that are updated in response to market participation events or resource events. Each virtual resource \( i \) is associated with two schedules, a power schedule, \( S_i^P \), and a capacity schedule, \( S_i^C \). If the aggregator service commits a certain resource to supply or consume power in the future, the power schedule of this resource is modified accordingly. If a resource is committed to only stand by for the provision of reserve, its capacity schedule is modified. If the reserve is activated, again the power schedule is updated.

Schedules are stored as lists of schedule fragments. Each schedule fragment \( F \) contains the ID of the corresponding virtual resource, \( i \), a bid ID \( k \) that associates the fragment with a previously placed bid, a start time \( t_i \), a stop time \( T_i \), and a power level \( P \), i.e., \( F = [i, k, t_i, T_i, p] \). Bid IDs \( k \) can either refer to energy market bids, then \( k \in K^E \), or to reserve market bids, then \( k \in K^R \).

Power profiles can be obtained from a list of schedule fragments by superposition (method sup) of individual schedule fragments. We define a power profile \( P \) as an ordered list of time intervals with corresponding power levels \( P_j \), i.e., \( P = \{P_1, P_2, \ldots\} \), where \( P_j = [t_j, T_j, p_j] \) and \( T_j = t_{j+1} \).

The aggregator service can query the database of resource schedules to obtain sets of resource availability statements, \( RASS = \{RAS_1, RAS_2, \ldots\} \). Each resource availability statement \( RAS \) consists of a resource identifier \( i \), the total change of the energy storage level that can be accessed during the requested time interval, \( \Delta_{\text{tot}} \in \{l_i - T_i, T_i - t_i\} \), where \( l_i \) and \( T_i \) denote the minimum and maximum energy storage levels of energy storage \( i \), and a set of availability intervals \( AIS = \{AI_1, AI_2, \ldots\} \) with \( AI_j = [t_j, T_j, p_j, T_j] \) describing when resources can be activated (between \( t_j \) and \( T_j \)) and at which power level (between \( p_j \) and \( T_j \)). Thus, a resource availability statement can be formally described as \( RAS = [i, \Delta_{\text{tot}}, AIS] \).

To obtain a resource availability statement from a resource, the following parameters have to be specified: a start time \( t_i \), a stop time \( T_i \), the current storage state of the resource, and the direction of the power flow from the resource perspective \( f \in \{in, out\} \). If \( f = in \), resource availability statements for supplying energy are returned; if \( f = out \), resource availability statements for consuming energy are returned.

In the following, we write \( X, y \) to denote the value of variable \( y \) in data set \( X \). All variables that are not explained in the nomenclature are defined in the algorithm itself. Variables \( t \) always denote times, variables \( d \) time durations, variables \( l \) energy storage levels, variables \( \Delta \) changes of energy storage levels, and variables \( p \) power levels. Constants, such as the parameters describing resource capabilities \( (l_i, T_i, p_i, \eta_i, c_{in}^i, c_{out}^i) \), as well as variables describing the current state of resources \( (t_i, T_i, p_i, T_i, c_i^P, c_i^C) \), are assumed to be known globally and therefore not provided as input to algorithms. The function \( S_{\text{next}}() \) allows for selecting the next element in an ordered set \( S \).

The algorithm of the method getRAS (cf., Figure 2),
which computes the resource availability statement for one resource, is listed in Figure 2. It first builds the power profiles $P^C$ and $P^D$ from the capacity and power schedules of the corresponding resource (lines 2&3). After initializing several variables (lines 4-7), it begins to traverse the power profiles while identifying time intervals with room to schedule further charging or discharging. For each considered time interval, the algorithm computes the current power level $p_{cur}$. Based on $p_{cur}$, the algorithm can determine if there is still room to charge or discharge (line 10). For each of these candidate intervals $[l_{cur}, t_{new}]$, it invokes method $getMinMaxL$ (cf., Figure 3) to compute the minimum and maximum energy level, $l_{cur}$ and $t_{cur}$, that could theoretically be observed until the end of the current candidate interval when starting at the current time $t_{now}$ and energy level $l_{now} + \Delta_{tot}$. The algorithm $getRAS$ (cf., Figure 2) always adds the identified candidate intervals $[l_{new}, t_{new}]$ to the set of availability time intervals $AIS$ (line 24). For each of these intervals, it also computes the corresponding change of the energy storage level $\Delta_{new}$ (lines 14-23) and updates the variable $\Delta_{tot}$ accordingly (line 26).

The algorithm of $getMinMaxL$ is listed in Figure 3 and described in further detail below. It simulates the development of the energy storage level of a given resource based on the two provided power profiles $P^C$ and $P^D$ starting with the initial energy level $l_{init}$ and returns the minimum and maximum levels it encounters after time $t_{search}$, $l_{tot}$ and $t_{tot}$. To determine the maximum energy level that could occur, the algorithm assumes that in addition to the scheduled discharging energy all possible discharging capacity is activated and none of the scheduled charging capacity is activated (lines 11&12).

### III. Optimization

The aggregator service has to approximate solutions of two types of optimization problems, denoted by $A$ and $B$. In the following, we propose three approximation methods. Two of the proposed methods are based on standard optimization techniques and are referred to as continuous optimization (Ct) and integer programming (IP). The third method is based on greedy scheduling (Gr). In the case of continuous optimization, the resource power level $p_{i,t}$ of a resource $i$ during time interval $t$ can be chosen from the continuous interval defined in the resource availability statements, while in the case of integer programming it can only be chosen from a fixed number of the predefined power levels, in particular $p_{i,t} \in \{\eta^{-1}p_{i}, 0, \eta p_{i}\}$.

All proposed optimization methods can be used to find approximate/heuristic solutions of the same optimization problems $A$ and $B$, namely computing maximum bid sizes based on the obtained resource availability statement (problem $A$, and computing a feasible dispatch schedule given a certain bid size considering resource priority. Whereas the IP and the Ct method presented in Section III-A allow for dividing the availability intervals $A_{i,j}$ contained in each resource availability statement into smaller activation durations of (minimal) size $d_{sched}$, the greedy scheduling algorithm (Gr) explained in Section III-B is less flexible: It greedily chooses resources that can be activated without considering more optimal allocations that could be possible if the division of availability intervals into smaller activation intervals. This leads to less optimal results with respect to energy brought to market, but can be done much faster (cf., Section IV-C).

#### A. Continuous Optimization and Integer Programming

The objective of optimization $A$ is to yield the maximum power level that can be maintained by a pool of resources during time interval $[t, T]$ based on the information available at time $t_{now}$ when the optimization is performed. The time duration $T - t$ must be divisible by $d_{sched}$ without remainder. The integer time horizon $h$ used in the following is equal to $(T - t)/d_{sched}$. The integer variable $m$ denotes the number of retrieved resource availability statements, i.e., $m = |RASS|$, where $RASS$ is the set of resource availability statements retrieved at the current time $t_{now}$. The decision variable $p_{i,t}$ is the power level of resource $i$ during time interval $t$. Equations 1 provide a formal definition of optimization problem $A$.

---

1. The capacity and power scheduled for a single resource can both be zero, both positive, or both negative: A single resource can never be scheduled to charge and discharge at the same time.
\[
\begin{align*}
\text{max} & \quad \sum_{i=1}^{m} |p_{i,1}| \quad (1a) \\
\text{s.t.} & \quad \sum_{i=1}^{m} |p_{i,t}| \in \left[ \sum_{i=1}^{m} |p_{i,1}| - \epsilon_M, \sum_{i=1}^{m} |p_{i,1}| + \epsilon_M \right] ; \\
& \quad p_{i,t} \in \left\{ \mathbf{AI}_j \right\} \quad \text{or} \quad p_{i,t} \in \left\{ \mathbf{AI}_j \right\} ; \\
& \quad \forall \mathbf{AI}_j \in \text{RAS}_i, \text{AIS} : t \geq \mathbf{AI}_j, \times \wedge t < \mathbf{AI}_j, \times ; i = 1, \ldots, m \\
& \quad 0 \leq \sum_{i=1}^{h} \lambda_i |p_{i,t}| d_{sched} \leq |\text{RAS}_i, \Delta_{tot}|; \quad (1d) \\
& \quad p_{i,t} = 0; \forall \mathbf{AI}_j \in \text{RAS}_i, \text{AIS} : t < \mathbf{AI}_j, \times \vee t \geq \mathbf{AI}_j, \times; \quad i = 1, \ldots, m \quad (1e)
\end{align*}
\]

Constraint 1b makes sure that the total power level over
the optimization horizon does not differ by more than 2\(\epsilon_M\)
between any two time intervals. In the case of continuous
optimization, the tolerance value \(\epsilon_M\) can be set to zero, whereas
in the case of integer programming, \(\epsilon_M\) has to be adjusted to the
properties of the resource pool. One possible choice is to set
\(\epsilon_M = \min_{i=1,\ldots,n} \left\{ \max \left\{ -\frac{p_i}{\eta_i}, \frac{p_i}{\eta_i} \right\} \right\} \). Constraint 1c defines the admissible value ranges of the decision variables when they are available for (dis)charging for the case of continuous optimization or integer programming, respectively. Constraint 1d assures that no resource supplies or consumes more energy than it can. The value of \(\lambda_i\) in Equation 1d is equal to \(\eta_i^{-1} c_i^{out}\) if a bid for negative reserve or energy purchase is computed, and equal to \(\eta_i^{-1} c_i^{out}\) if a bid for positive reserve or energy sale is computed. Constraint 1e makes sure that a resource is only activated within the time periods during which it is available.

The result of optimization A is a maximum bid size, \(p_{\text{max}}\), which is positive in the case of energy supply (\(f = \text{out}\)), and negative in the case of energy consumption (\(f = \text{in}\)). Since optimization A does not consider any restrictions regarding the number of times that a resource can be switched on and off, it yields an upper bound of the achievable power level based on resources states at the current time \(t_{\text{now}}\). The value of \(p_{\text{max}}\) will most likely not conform to the bid size constraints of the target market. However, one can show that every bid size \(p \in [0, p_{\text{max}}]\) can be achieved using the same set of resource availability statements. We exploit this property to speed up the resource management processes.

The objective of optimization B is to determine feasible controls for achieving a certain power level \(p_t \leq p_{\text{max}}\). Furthermore, it allows for specifying an arbitrary objective that can be used to prioritize the use of certain resources. The function \(\text{prio}(\text{RAS}_i)\) returns a higher value for a higher priority of including resource \(i\) into the schedule. Equations 2a - 2e formally describe the optimization problem to be solved.
if \( f = \text{in} \) then

\[
\lambda_i \leftarrow \eta^{m} \cdot p_i \leftarrow \text{RASS}_{A_i}.p_i,
\]

else if \( f = \text{out} \) then

\[
\lambda_i \leftarrow \eta^{m} \cdot p_i \leftarrow \text{RASS}_{A_i}.p_i,
\]

if \( < \text{h} \) then

\[
\text{if } RAS_{A_i}.t \geq \text{last}_{A_i} + \text{RAS}_{A_i}.t_{\Delta} \geq \lambda_i | p_i | d_{\text{sched}} \text{ then}
\]

\[
p_t \leftarrow p_t + | p_i |,
\]

\[
\text{RAS}_{A_i}.t \leftarrow \text{RAS}_{A_i}.t - \lambda_i | p_i | d_{\text{sched}}
\]

else

\[
\text{opt} \leftarrow B
\]

\[
F_{\text{new}} \leftarrow \{ \text{RAS}_{A_i}.j \leftarrow \{ i, \lambda_i, p_i \} \};
\]

\[
F_{\text{out}} \leftarrow F_{\text{new}} \cup F_{\text{new}}
\]

17: \text{RASS}_{\text{act}} \leftarrow \text{RASS}_{\text{act}} \setminus \text{RAS}_{A_i};

18: \text{if } RAS_{A_i}.t < \text{last}_{A_i} \land \text{RAS}_{A_i}.t_{\Delta} \geq \lambda_i | p_i | d_{\text{sched}} \text{ then}

19: \text{if } j_i < | \text{RAS}_{A_i} | 1

20: \text{RAS}^{+} \leftarrow \text{RASS}_{\text{in}} \cup \text{RAS}_{A_i}; j_i \leftarrow j_i + 1

21: \text{if } = \text{h} \) then

22: \text{if } RAS_{A_i}.t \geq \text{last}_{A_i} \land \text{RAS}_{A_i}.t_{\Delta} \geq \lambda_i | p_i | d_{\text{sched}} \text{ then}

23: \text{if } opt = B

24: \text{F}_{\text{new}} \leftarrow \{ \text{RAS}_{A_i}.j \leftarrow \{ i, \lambda_i, p_i \} \}

25: \text{F}_{\text{out}} \leftarrow \text{F}_{\text{new}} \cup \text{F}_{\text{new}}

26: \text{if } pt < \text{target} - \epsilon_M \text{ then}

27: \text{RASS}_{\text{sched}} \leftarrow \text{selRASS}(\text{RASS}_{\text{in}}); I_1

28: \text{if } RASS_{\text{sched}} \neq \emptyset \text{ then}

29: \text{RAS}_{A_i} \leftarrow \text{first}(\text{RASS}_{\text{sched}})

30: j_i \leftarrow \text{getAIIIndex}(\text{RAS}_{A_i}, I_1)

31: \text{if } \text{in} \text{ then}

32: p_t \leftarrow \text{RAS}_{A_i}.p_i

33: \text{else if } f = \text{out} \text{ then}

34: p_t \leftarrow \text{RAS}_{A_i}.p_i

35: \text{while } pt + \left| p_i \right| < \text{target} - \epsilon_M \text{ do}

36: \text{RAS}_{A_i}.j \leftarrow \{ i, \lambda_i \}

37: \text{RASS}_{\text{sched}} \leftarrow \text{RASS}_{\text{sched}} \setminus \text{RAS}_{A_i}

38: \text{RASS}_{\text{in}} \leftarrow \text{RASS}_{\text{in}} \setminus \text{RAS}_{A_i}

39: \text{RASS}_{\text{act}} \leftarrow \text{RASS}_{\text{act}} \setminus \text{RAS}_{A_i}

40: \text{if } \text{h} \text{ then}

41: \text{if } opt = B

42: \text{F}_{\text{new}} \leftarrow \{ \text{RAS}_{A_i}.j \leftarrow \{ i, \lambda_i, p_i \} \}

43: \text{F}_{\text{out}} \leftarrow \text{F}_{\text{new}} \cup \text{F}_{\text{new}}

44: \text{if } pt < \text{target} - \epsilon_M \text{ then}

45: \text{RAS}_{A_i} \leftarrow \text{getFirst}(\text{RASS}_{\text{sched}})

46: j_i \leftarrow \text{getAIIIndex}(\text{RAS}_{A_i}, I_1)

47: \text{if } \text{in} \text{ then}

48: \text{RASS}_{\text{sched}} \neq \emptyset \text{ then}

49: p_t \leftarrow p_t + | p_i |

50: \text{F}_{\text{out}} \leftarrow \text{F}_{\text{out}} \cup \text{F}_{\text{new}}

51: \text{if } RASS_{\text{sched}} \neq \emptyset \text{ then}

52: \text{RAS}_{A_i} \leftarrow \text{getFirst}(\text{RASS}_{\text{sched}})

53: j_i \leftarrow \text{getAIIIndex}(\text{RAS}_{A_i}, I_1)

54: \text{if } \text{in} \text{ then}

55: p_t \leftarrow p_t + | p_i |

56: \text{else if } f = \text{out} \text{ then}

57: p_t \leftarrow \text{RAS}_{A_i}.p_i

58: \text{else}

59: \text{break}

60: \text{if } pt < \text{target} - \epsilon_M \text{ then}

61: \text{return false}

62: \text{if } opt = B

63: \text{return \emptyset}

64: \text{return true}

65: \text{if } f \text{ then}

66: \text{return } F_{\text{out}}

Fig. 4. Greedy scheduling algorithm \text{grSched} for approximating a feasible solution of optimization problem \( B \).
IV. SIMULATION STUDY

To evaluate the proposed system features, we conducted a set of simulation experiments based on a dual market use case described in Section IV-A. The setup of these experiments is described in Section IV-B, the results in Section IV-C. The use case is not exhaustive in terms of evaluating different bidding strategies and possible energy storage options, but was chosen carefully to show how the algorithms described above would perform while applying a basic but profitable bidding strategy in a realistic market environment. Our goal is thus not to propose a financially optimal market bidding strategy or to identify the lowest cost energy storage option. However, the functionality of the aggregator service can be leveraged in future work to implement different bidding strategies and controlling a different set of LERs.

A. Use Case

We consider a dual market use case, where an aggregator concurrently participates in a market for negative reserve RM and an energy spot market EM. The aggregator’s primary objective in this use case is to offer as much reserve as possible in the long run. Since the activation of reserves is highly uncertain and therefore hard to predict, one possible bidding strategy is to keep the total energy level of the resources in the pool at a minimum. The aggregator therefore try to sell as much stored energy as possible on the energy spot market in the shortest time possible. This resource management policy calls for a greedy bidding strategy: The aggregator bids for the highest level of reserve it can provide in subsequent reserve market intervals and also bids for selling the highest amount of energy in subsequent spot market intervals. On the revenue side, this strategy results in three revenue streams: the revenue earned per MW of negative reserve sold on the market (standby revenue), the per MWh revenue obtained when the reserve is activated (activation revenue), and the revenue earned by selling energy on the energy market (energy restoration revenue). Considering this use case allows us to evaluate the coordination of two resource management processes accessing the same set of resources.

We assume that the aggregator has full control over a pool of n residential size vanadium redox batteries (VRBs). VRBs are flow batteries, i.e., they take advantage of the electric potential between two chemical components dissolved in liquids. The main advantages of VRBs are their long life time and easy scalability — since they do not contain electrodes like lead acid or lithium ion batteries, they cannot degrade chemically. However, their potential has to be maintained by mechanical pumps, which have to be maintained. The energy storage capacity of a VRB can be scaled by simply adding more
dissolved vanadium to its tanks. Since the main disadvantage of VRBs, namely a relatively low energy density compared to other battery chemistries, does not play a major role in stationary applications, we expect an increasing use of this technology for the purposes discussed in this article. Using a different battery chemistry, in particular traditional chemical batteries, may require that the computation of resource availability statements presented in Section II-B takes the gradual decline of energy storage capacity resulting from chemical degradation explicitly into account. This could be done by changing the individual resource parameters that are affected by chemical degradation, e.g., \( \frac{1}{l_i} \) and \( \frac{1}{l_i} \), based on their actual usage profiles.

Control signals destined for single storage resources are automatically created from the service’s power schedules. In this use case, we assume that the virtual resources representing these resources on the service level model their behavior perfectly, i.e., their energy capacity, power output, and charging/discharging behavior is known ex ante.

To obtain realistic VRB behavior, we have implemented the VBR model described in [7]. It assumes a 39 series cell stack, a nominal power rating of 3.3 kW, and an energy content of 9.9 kWh. The VRB’s discharge voltage range is 42-55 V, the discharge current range is 10-80 A. The energy storage level of each VRB is allowed to vary between 20 and 80%. For the VRB model we determined a charging factor \( c^\text{in} = -7,244 \times 10^{-10} \), and a discharging factor \( c^\text{out} = -10,045 \times 10^{-10} \). The model parameters provided in [7] were obtained from an actual VRB, therefore the model output should closely match the behavior of a real VRB. To simulate the supply and consumption at constant power levels, we implemented constant power point tracking: It adjusts the charging or discharging current of the battery in response to voltages at different states of charge. We also assume a 10% power loss when charging and discharging, i.e., \( \eta = 0.9 \), which accounts for realistic efficiency levels of single-phase rectifiers and inverters rated at the VRB’s nominal voltage and current.

B. Setup

For simplicity, we assume that both markets clear in time intervals of \( d^\text{RM} \) and \( d^\text{EM} \), respectively. Thus, bidding can take place at least once per operating interval. Single intervals of reserve activation last for \( d^\text{RM} \) and we assume that the entire reserve power is activated.\(^2\) Figure 6 shows the timing of simulated activities using an example. The chosen parameters in the example correspond with the default parameter configuration of our experiments.

The aggregator follows a greedy bidding strategy on both markets. It continuously places the highest possible bids offering negative reserve and the highest possible bids offering excess energy previously absorbed by the VRBs on the energy market, both before the corresponding bidding deadlines, respectively. For the use case, we used the following

\(^2\)This is a realistic assumption: Since TSOs usually activate reserve in ascending order of delivery prices, it is most likely that single providers of reserve, especially if they are rather small market players, have to provide their entire capacity instead of only a fraction of it.
prioritization function $\text{prio}(\text{RAS})$: Every time it is called, it shuffles the list of $m$ resource available statements and then assigns increasing integer priorities, i.e., 1, ..., $m$, to the resulting list. The idea behind this particular prioritization function is to minimize switching frequency while using resources as equally as possible in the long run. In fact, many types of priority functions are conceivable, e.g., functions that minimize resource utilization costs by assigning the inverse of the individual utilization cost of each resource as priority.

Reserve activation is simulated using independent random numbers $x_j$ drawn from a Bernoulli distribution $X(\gamma_{act})$. The simulation time increment is set to $d_{\text{sched}} = 5$ minutes, which is sufficient to capture the full detail of the scenario based on our assumptions. At the beginning of each simulation run, the state of charge of all VRBs is set to 20%, i.e., the minimum value. Thus, if the scheduling algorithm does not find a solution in the first reserve provision process, the pool size is too small to participate in the markets under the given restrictions.

Table I contains the parameter values we used for different simulation runs, where default values are indicated by *. We selected these values to match typical market parameters in Central Europe, where the European Energy Exchange (EEX) runs an intraday spot market which continuously and closely 45 minutes before each 15 minutes long operating interval [4]. The primary and secondary reserve markets in Germany currently clear on a weekly basis, the markets for tertiary minute reserve are currently 4 hours long. Minimum bid sizes for primary reserve are 1 MW, and 5 MW both for secondary and tertiary reserve. Bid size increments are currently 1 MW each. Reserve activation durations are 15 minutes for secondary and tertiary reserve. Thus, while we consider actual constraints of participating in the intraday energy market, our assumptions anticipate a situation where reserve markets are more flexible with respect to bid sizing and allow for more frequent participation. We used smaller bid increments and operating intervals because it facilitates revealing the impact of changes of these parameters on the chosen performance metrics. We set the time horizon of the simulation to 2 months, which is sufficient to collect the necessary empirical distributions shown in Section IV-C. The random number generator we used to determine reserve activation was initialized with the same seed for all simulation runs so that the aggregator service saw the exact same reserve activation periods across different runs.

The simulation was implemented as a single-threaded Java program. During code execution, all required Java objects were kept in main memory to avoid that hard drive access influenced performance. The performance results presented in the following section can thus be considered as a realistic preview of how an actual system would perform if all data required for the optimization processes can be kept in main memory.

All integer programs were solved using IBM ILOG CPLEX version 12.5 [8], which was invoked via its Java interface. Simulation programs were run on a server equipped with Intel Xeon E5-2650 processors (2.0 GHz). We allocated 10 GB of main memory to each program, which was sufficient even for the largest instances ($n = 5, 000$).

Time measurements were carried out using native Java methods, which are sufficiently accurate given the relevant time scales (milliseconds to seconds). The simulation code was run on a dedicated server making sure that each program could use exactly one core at full capacity and was able to access ample main memory. We measured the time of entire service method executions, e.g., the time it takes the service to compute the maximum reserve bid size for a future time interval, to account for all necessary data processing.

### C. Results

1) Market Performance: Figure 7 shows the average amount of energy brought to the energy market each day.

![Figure 7](image)

**Fig. 6.** Timing of simulated activities

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>500; 1,000; ...; 5,000</td>
<td>Number of resources</td>
</tr>
<tr>
<td>$d_{\text{sched}}$</td>
<td>5 min</td>
<td>Duration of optimization time intervals</td>
</tr>
<tr>
<td>$d_{\text{act}}$</td>
<td>15 min</td>
<td>Duration of reserve activation for $RM$</td>
</tr>
<tr>
<td>$d_{\text{bid}}$</td>
<td>2; 4*; 6 h</td>
<td>Duration of $RM$ operating interval</td>
</tr>
<tr>
<td>$d_{\text{EM}}$</td>
<td>1 h</td>
<td>Time between $EM$ bidding deadline and start of operating interval</td>
</tr>
<tr>
<td>$p_{\text{EM}}$</td>
<td>0.25; 0.5*; 0.75 MW</td>
<td>Minimum $EM$ bid size</td>
</tr>
<tr>
<td>$p'_{\text{EM}}$</td>
<td>0.25; 0.5*; 0.75 MW</td>
<td>Minimum $EM$ bid size increment</td>
</tr>
<tr>
<td>$\alpha_{\text{RM}}$</td>
<td>0.1; 0.3*; 0.5</td>
<td>Probability of $RM$ calls</td>
</tr>
<tr>
<td>$d_{\text{EM}}$</td>
<td>15 min</td>
<td>Duration of $EM$ operating interval</td>
</tr>
<tr>
<td>$d_{\text{bid}}$</td>
<td>1 h</td>
<td>Time between $EM$ bidding deadline and start of operating interval</td>
</tr>
<tr>
<td>$p_{\text{EM}}$</td>
<td>0.25; 0.5*; 0.75 MW</td>
<td>Minimum $EM$ bid size</td>
</tr>
<tr>
<td>$p'_{\text{EM}}$</td>
<td>0.25; 0.5*; 0.75 MW</td>
<td>Minimum $EM$ bid size increment</td>
</tr>
</tbody>
</table>

*This assumption allows us to assess the impact of random activation without having to restrict the evaluation to a specific market scenario.*

*Although CPLEX is able to use multiple cores, we intentionally restricted it to use only one core at once.*
the provided negative reserve expressed in (potential) energy per day, and the activated reserve for different pool sizes based on the default parameter configuration provided in Table I. For both optimization methods, these values increase or decrease in a slightly superlinear way. There are some signs of economies of scale, but they are rather small and decreasing. Thus, the aggregator service could divide the managed resources into groups of equal size and perform the optimization separately for each group without limiting the total energy brought to market.

The results shown also reveal that the solutions based on continuous optimization and integer programming yield approximately equal performance and always beat the greedy scheduling based method. This is not surprising since these approaches allow for switching batteries on and off more frequently, which allows for allocating smaller fragments. This feature lets the aggregator service build larger blocks using continuous or integer optimization compared to greedy scheduling based optimization even if total available energy represented by the resource availability statements is equal.

The box plots of Figure 8 show that the distribution of reserve market bid sizes produced by the continuous and integer optimization approaches are also approximately equal. For 5,000 VRBs, the aggregator can offer up to 12 MW of negative reserve 75% of the time, and above 8 MW 99% of the time using the IP approach. Thus, the analysis of our use case also demonstrates that a pool of batteries can sustain a stable minimum level of reserve over time, similar to conventional power plants today. The greedy scheduling based approach leads to less predictable bid size distribution, in particular with respect high bid sizes. This outcome can again be explained by the limitations of the greedy scheduling approach: It is less flexible with respect to resource allocation during bid computation.

The considered changes of the minimum bid sizes and bid size increments (+/-0.25 MW) only have a small impact on energy brought to market. Furthermore, the gap between the continuous/IP-based optimization and the greedy scheduling approach remains approximately equal in these cases.

The probability of reserve activation has a negative impact on the advantage of the continuous and IP-based optimization approaches over the greedy scheduling based approach. Generally, the more of the reserve gets activated on average, the smaller the amount of average reserve committed, since it takes longer to empty the batteries until the next reserve market bidding deadline.

2) Resource Management Performance: A more detailed view of resource utilization allows us to gain insights regarding the efficiency of the proposed control methods. The more energy chemical batteries have to charge or discharge and the more often they have to change their power state, the faster they usually degrade. When managing resources in a pool, it would also be advantageous to distribute the burden as equally as possible across all resources.

Regarding execution time, the faster the service can respond to aggregator requests, the more opportunities the aggregator has to maximize resource utilization and market revenue by evaluating different trading strategies based on the current status of the resource pool. A fundamental requirement is that

5The remaining market energy metrics shown in Figure 7, reserve activated and energy sold, are proportional to this metric.
resource management processes do not take longer than the time available, i.e., the time required to compute maximum bids plus the time to compute final bids must not exceed the time left before the bidding deadlines of the corresponding markets. Furthermore, the time to activate reserve must not exceed the maximum activation delay required by the system operator.

As Figure 12 shows, the latency experienced by the aggregator when requesting maximum reserve market bid sizes, i.e., solving optimization problem $A$, are significantly lower if the greedy scheduling based approach is applied compared to the other approaches. For a pool size to $n = 5,000$ batteries, the highest observed time to compute the maximum bid size is approximately 2 seconds if using the greedy scheduling approach, compared to 13 and 16 seconds if integer programming or continuous optimization is used, respectively. While the time to compute maximum bid sizes usually increases exponentially with the number of controlled resources across all considered optimization approaches, the integer programming approach requires much longer at times (cf., results for $n = 1,500; 2,000; 2,500$ in Figure 12), which is due to the unpredictable behavior of IP solving heuristics.

Figure 13 shows the system’s speed of committing virtual resources according to a given feasible power level, i.e., solving optimization problem $B$. In case of continuous optimization and integer programming, this process takes approximately twice as long as maximum bid computation. The greedy scheduling based optimization allows for a significantly faster commitment of virtual resources since $getGrSched$ (cf., Figure 4) needs to be invoked exactly once, albeit in schedule generation mode.
The final dispatch of virtual resources does not involve optimization and can therefore be done much faster than the computation of the maximum bid size or committing resources. Still, the greedy scheduling approach outperforms the other approaches regarding this metric, as well. Since schedules are less fragmented in this case (fewer possibilities to switch), the process of transferring schedule data from the capacity to the power schedules is less time-consuming.

Fig. 14. 99.5th percentile duration for computing maximum reserve bids (optimization type A)

Fig. 15. 99.5th percentile duration for computing final reserve bids (optimization type B)

Figures 14 and 15 show how the worst case response times for computing maximum reserve bid sizes (optimization problem A) and final resource allocations for fixed bid sizes (optimization problem B) are influenced by changes of the different sensitivity parameters considered in this study. The results indicate that across all considered parameter configurations, continuous optimization requires the most time, followed by integer programming and the greedy scheduling approach. On average, the greedy scheduling approach is 10 to 20 times faster than the other approaches. The observed performance gaps increase with the duration of the reserve market operating interval and the reserve activation probability. The considered range of bid size constraints only has a minor impact on these performance differences.

V. Discussion

A. Summary of Contributions

We have proposed a detailed service logic that enables concurrent multi-market participation of large numbers of energy storage devices, such as the small VRBs considered in our use case. The flexibility required by this task complicates the use of simple control schemes, e.g., dispatching all energy storage devices proportionally to mimic one large energy storage (cf., Section V-B). Organizationally, the proposed market participation is mediated by an aggregator, a company specializing in the control of distributed storage or other LERs to achieve a superordinate target, e.g., providing as much reserve as possible as in our use case. A central contribution of this work is the explicit consideration of market participation constraints and the computational complexity resulting from continuously monitoring and controlling many resources: In contrast to the related work presented in Section VI, we consider the requirements of typical bidding processes used by contemporary wholesale electricity markets. These include deadlines for placing bids that precede the start of the corresponding operating intervals, minimum bid sizes and bid increments, as well as fixed operating intervals. Furthermore, we developed effective algorithms for on demand computation of so-called resource availability statements. These data sets describe the capability of a storage device to deliver reserve or supply/consumer energy during a given future time period, taking its current energy storage level, individual constraints, and the already scheduled future activities into account. The market and resource models we use are realistic and, given the existence of a reliable two-way communication channel between service layer and the physical resource, the evaluated control strategy could be implemented in practice.

B. Comparison with Sequential Proportional Storage Control

If the energy storage devices controlled by the aggregator allow for continuous power dispatch, a simple dispatch policy would be to control the $n$ batteries as one large energy storage. In this case, all batteries would be dispatched proportionally and always have the same state of energy storage over time. This type of control does not require any optimization, but limits the flexibility of the aggregator. In particular, concurrent participation in several markets would not be possible and the market constraints would limit resource utilization. We provide the following example to back these insights.

The VRBs considered in our use case can charge for up to 4.2 hours at full power: $(I_i - L_i)/(c_i \times P_i) = 0.6/(-7,244 \times 10^{-10} \times -3,300) = 251$ min $\approx 4.2$ h. Therefore, with $n = 5,000$, the aggregator could provide negative reserve of up to $n \times c_i / P_i = 5,000 \times 0.91 \times -3,300 \times 10^{-6} \approx -18.33$ MW over 4 hours. When considering the default bid size increment constraint, i.e., $p_{\text{min}}^R = 0.5$ MW, the maximum bid provided that all batteries are empty is 18 MW. Bids could be placed in 3 out of the total 24/4 = 6 reserve market operating intervals available every day, i.e., the total amount of reserve provided would be $3 \times 4 \times -18 = -216$ MWh/day. During the 4 hours left in between the reserve market operating intervals, the aggregator would sell energy on the
spot market to empty the batteries. The round trip efficiency of the considered VRBs is \(c^{in}/c^{out} = 0.721\) and the default long-run reserve activation probability is \(\gamma_{RM} = 0.3\), therefore the energy sold would amount to \(0.9 \times 0.721 \times 0.3 \times 216 \approx 42\) MWh/day. These values can now be compared to our results, which show a maximum of -311 MWh of reserve and 54 MWh of energy sold for \(n = 5,000\) (cf., Figure 7). Although these values differ for other parameter configurations, this short example demonstrates the value of our approach compared to simply dispatching \(n\) batteries like one big battery, even without the additional flexibility that our approach is able to provide.

C. Economics of the Use Case Bidding Strategy

In the presented use case, the aggregator favors the supply of negative reserve over selling energy on the spot market, which implies that it expects higher revenues by providing reserves than by spot market arbitrage. Although our goal in this article is not to propose financially optimal bidding strategies, we offer the following calculation to back this choice.

All figures used in the following are based on market data collected over the course of 2 years (2012 & 2013), which is publicly available at [5] and [4]. The prices on the intraday spot market in Germany range between approximately 0 and 100 EUR/MWh (99% confidence interval). The average observed spot market price is approximately 40 EUR/MWh. Typical daily minimum and maximum prices are 20 and 80 EUR/MWh, respectively.

Thus, given the parameters of our use case and assuming daily high and low prices last for one hour, the aggregator could make a maximum of \(5,000 \times 0.9^2 \times 0.721 \times 3,300 \times 10^{-6} \times 60 \approx 578\) EUR per day by charging at the minimum price and discharging at the maximum price. Average standby prices for negative secondary reserve range between 660 and 1,287 EUR/MW for day- and night-time weekly tenders. Current average prices range at 11.5 EUR/MWh. Average activation prices range between 0 and 821 EUR/MWh, but only start to rise above zero at activation levels of greater than 1,000 MW, which only happens 5% of the time. Thus, the revenue the aggregator could gain via activation fees is rather small and we therefore neglect it in this calculation. If the aggregator applied sequential proportional storage control (cf., Section V-B), it could make \(216 \times 11.5 = 2,484\) EUR/day on the reserve market and \(42 \times 40 = 1,680\) EUR/day on the energy spot market, i.e., a total of \(4,164\) EUR/day. Using the system proposed in this article, the aggregator could earn a total of \(311 \times 11.5 + 54 \times 40 = 5,736.5\) EUR/day, i.e., approximately 38% more. Given these results, ranking the provision of reserves over spot market arbitrage is a reasonable decision.

Despite the fact that providing reserves is more profitable than energy spot market arbitrage, the basic greedy bidding strategy applied in our use case still has potential for improvement. For instance, spot market bids could be placed more selectively to maximize the revenue from selling, i.e., the variability of spot market prices could be explicitly considered. However, waiting longer to restore the low energy storage levels needed to offer negative reserve could affect the revenue gained via reserve standby, which is priced almost as high on average as energy traded at spot market price peaks. Moreover, as can also be seen from the exemplary calculations above, the spot market revenue is only a relatively small fraction of the total market revenue (\(\approx 10\%\)) in the use case. Thus, we expect the economic benefit gained from financial optimization of use case bidding strategy to be rather small.

D. Comparison of Optimization Approaches

Our comparison of several optimization methods for computing wholesale market bids from resource availability statements has revealed their complementary strengths: The continuous and integer programming optimization approaches achieve better resource utilization in terms of energy brought to market, whereas the greedy scheduling based approach is faster and causes less power state changes. Our sensitivity analysis has shown that these relationships can change based on the market rules. In particular, the duration of market operating intervals and the reserve activation probability have a significant influence on the energy brought to market and the time to compute market bids. Therefore, the optimal choice of optimization methods depends on market rules, bidding strategy, and resource capabilities/costs. They could also be combined, e.g., to obtain a preliminary but less optimal solution using the greedy scheduling based approach first while concurrently searching for a better solution using the other methods.

E. System Scalability

Our results indicate that the control of a larger pool of resources can be split up into smaller subgroups without limiting the total amount of energy brought to market (cf., Figure 7). Thus, service requests could be processed on several servers in parallel, providing more timely results. In particular, subgroups of the total population of virtual resources could be allocated to individual optimization engines that would each compute bid allocations for their subgroup. The aggregator service could then report the sum of the bid sizes reported by the individual optimization engines to the aggregator application. Faster computation of resource allocations enables more complex bidding strategies, which would involve repeated invocations of optimization routines before the bidding deadline.

F. Limitations and Future Work

The evaluation of the proposed system design could be extended to many more bidding strategies and types of LERs. Regarding bidding strategies, aggregators could apply other types of bidding strategies, such as to provide positive reserve while buying energy on the spot market or to participate in markets for positive and negative reserve. Furthermore, aggregators could apply methods to explicitly maximize profits based on price forecasts and combine them with the functionality offered by the proposed system features. In particular, the decision which bids should be placed on which market when could be made with the goal of maximizing long-term profits.
This article contributes to enabling the control of many energy storage devices in accordance with market rules, but does not propose profit maximization schemes. Regarding other types of LERs, the solutions presented in this article focus on dedicated and fully controllable devices, in particular stationary battery energy storage. We believe, however, that the proposed approach could also be a good starting point for controlling LERs with uncertain availability, e.g., electric vehicles.

VI. RELATED WORK

A number of recent publications investigate the provision of reserves by energy storage and flexible loads [9], [3], [10], or their direct coupling with renewable power generation [11], [12], [13]. Market participation scenarios have been investigated, among others, by the authors of [14], [15], [10], [16], [17], and [18].

In [14], the authors investigate the market value of electric vehicles and demonstrate that using them for providing reserves can be a promising business case. In contrast to them, our work deals with the challenge of controlling large numbers of LERs according to market constraints and we also focus on stationary energy storage rather than electric vehicles.

Similar to [14], the purpose of the work presented in [15] is to gauge the business value of vehicle-to-grid reserves in Germany. Their conclusion is that providing negative secondary reserve over night yields the highest economic benefit. Although they consider more realistic reserve market constraints than the other papers cited in this section, they do not consider individual resource control or concurrent market participation. Instead, they divide the expected EV charging energy at time \( t \) by a fixed duration of 1-4 hours, taking the minimum of the resulting values during night times as the maximum level of reserve that the EVs can provide.

A more recent article, [10], proposes approximate dynamic programming for maximizing the value of energy storage by using it for different purposes, including the participation in electricity markets. However, the authors of [10] do not consider actual market rules and focus on finding optimal dispatch policies for one battery, not large pools of batteries. The authors of [15] do consider the use of many electric vehicles, but they do not consider the challenge of dispatching them such that aggregators can actually provide MWh scale blocks of secure reserve like us.

Similar to us, the authors of [19] investigate how operators of energy storage might derive profits from absorbing variable power demand, e.g., from activated reserve, while trading energy at the same time. They apply stochastic dynamic optimization to derive a number of theoretical results based on several simplifying assumptions, in particular that optimal management policies for single storages can be obtained from known unscheduled power demand characteristics and future energy prices, if one assumes that unscheduled power demand is temporally uncorrelated and energy transactions are perfectly efficient. On the one hand, we use less sophisticated mathematical models than [19] and do not make the details of the mathematical optimization process explicit (we merely formulate integer programs and hand them over to CPLEX for finding feasible solutions). On the other hand, our problem formulation enables us to consider realistic market participation constraints, such as the guaranteed ability to deliver reserve, minimum bid sizes, certain bid size increments, and temporal bid constraints. We also conduct experiments to demonstrate the computational scalability in a realistic use case.

The authors of [9] propose innovative methods for controlling large pools of thermostatically controlled loads to manage real-time energy imbalance. Although they consider tracking a regulation signal instead of a market participation scenario like us, their work is related to ours as it investigates how control of large pools could be realized without perfect information. Our approach, however, requires perfect information about resource state.

In [12], the authors develop and analyze real-time scheduling algorithms for coordinating deferrable loads for reserve provision. They do not consider a general market participation scenario but instead investigate how different scheduling approaches can be used to match deferrable load to variable solar generation output, i.e., a direct coupling scenario. Direct coupling is particularly attractive in a scenario where the power system operator can directly control deferrable loads. They compare two real-time scheduling policies, earliest deadline first and least laxity first, against a receding horizon control (RHC) approach. In their RHC approach, a quadratic continuous optimization problem is solved for each 10 minutes interval within a 24 hour period. The optimization goal of the RHC is to minimize reserve energy and/or capacity. Thus, although the authors of [12] are also interested in how one can schedule large numbers of energy delivery tasks to match an aggregate profile in as little time as possible, their concrete scenario, assumptions and tools differ from ours, which makes it impossible to compare the quantitative results.

Additional related work that treats the control of small-scale energy storage and other types of LERs in power systems, albeit with a different focus than this article, includes [20], [21], [22], [23], and [24].

VII. CONCLUSIONS

We have proposed and evaluated efficient abstractions and algorithms that can be used to realize a service layer between aggregators and pools of energy storage devices. The proposed service logic enables aggregators to control large numbers of energy storage devices such that they can participate in several wholesale electricity markets at the same time. Based on a realistic and meaningful dual market use case, we have shown that the service fulfills this task in accordance with all relevant requirements. While the proposed continuous and discrete optimization approaches performs better than the proposed greedy scheduling based approach with respect to resource utilization, the latter approach has important advantages, as well: It is much faster and allows for much smaller battery switching frequency. The almost linear development of total energy brought to market with increasing pool sizes indicates that our system could scale to very large pool sizes if computations for resource subsets are performed in parallel.
We believe that our work is useful input for companies in the business of bringing distributed energy storage resources to market, e.g., demand response service providers. It provides guidelines for the development of flexible and scalable backend logic, which is capable of carrying out all necessary input/output translation tasks between markets and resources and allows for concurrent participation in several markets based on arbitrary bidding strategies. It also underlines the influence of market design decisions, such as minimum bid sizes and increments, timing constraints, and operating interval durations, on the revenue potential of such businesses: The more flexible these markets become, the more efficiently aggregators can dispatch pools of energy storage resources.

REFERENCES


