Capturing Battery Flexibility in a General and Scalable Way Using the FlexOffer Model

Fabio Lilliu, Torben Bach Pedersen, Laurynas Šikšnys

Aalborg University

October 12, 2021

Flexibility

Flexibility is highly important for employment of renewable energy sources.

There are precise mathematical models to describe flexibility, such as Linear Time Invariant (LTI) series.

However, these exact models take too much time for optimization and aggregation.

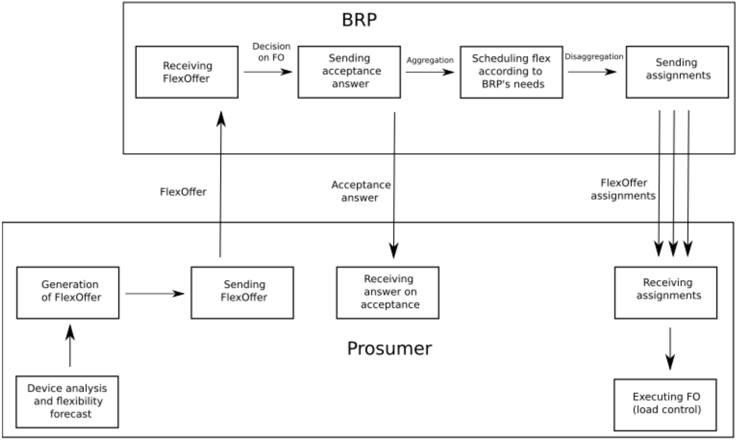
FlexOffers

As a solution, we propose FlexOffers (FOs): approximate models, but easy to optimize and aggregate.

We will:

1. show how to use FOs to model battery flexibility
2. show specific FOs generation algorithms
3. propose a faster, analytic algorithm for FO generation
4. define an exact model for aggregation in our experiments
5. show how FOs retain most of the flexibility while scaling a lot better than exact models

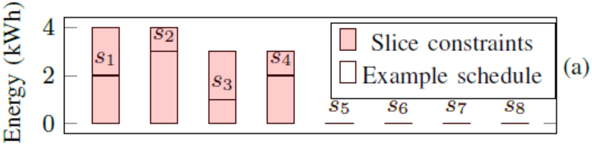
FlexOffers



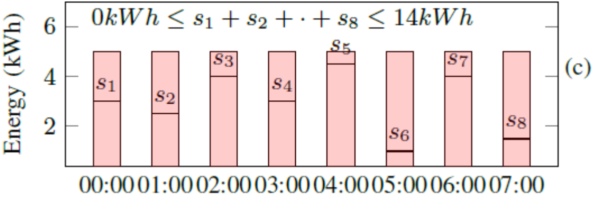
A FO is a set of constraints approximating the flexibility for a certain device. The figure represents the life cycle of a FO.

Types of FlexOffer

For a **standard FO** (SFO), we have *slice constraints* which indicate how much energy can be consumed by the device at each time unit.

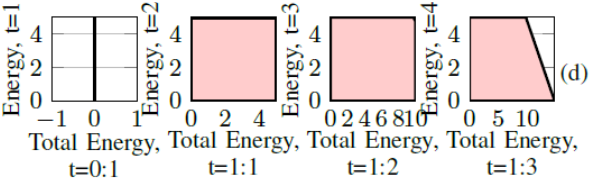


For a **total energy constraint FO** (TECFO), we have one additional constraint on the total amount of energy used by the device.



Types of FlexOffer

For a **dependency energy constraint FO** (SFO), the constraints determines how much energy can be used from the device depending on the amount of energy consumed until that time.



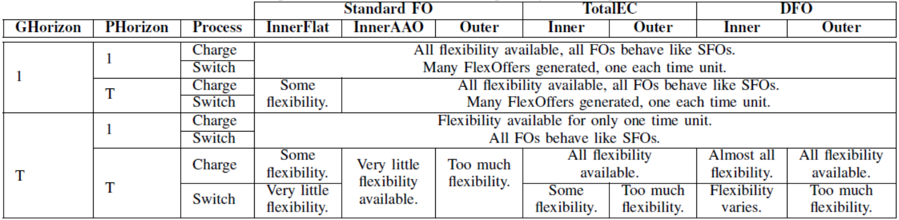
We have built *inner* and *outer* approximations of flexibility. In the former, flexibility is strictly less than the actually available amount; in the latter, strictly more. (Insert figure for explanation)

Parameters for FO generation

Generation of FOs is tailored on the specific case of the prosumer and battery. We identify the specific case through these parameters:

- **Type of constraints**: which constraints we want to use for the FOs (slices, TEC, DFO).
- **Allocation strategy**: For SFOs: allocating flexibility all at once, or in an uniform way.
- **Planning horizon (T)**: number of FO slices.
- **Generation horizon (N)**: how often FOs are generated.
- **Process type**: only charging/discharging the battery, or being able to switch at each time unit.

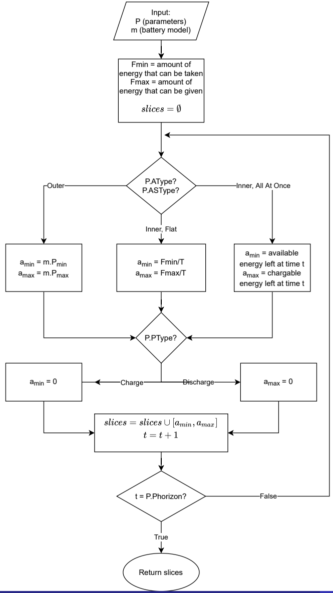
Use cases



			Standard FO			TotalEC		DFO	
GHorizon	**PHorizon**	**Process**	**InnerFlat**	**InnerAAO**	**Outer**	**Inner**	**Outer**	**Inner**	**Outer**
1	1	Charge	All flexibility available, all FOs behave like SFOs. Many FlexOffers generated, one each time unit.						
		Switch							
	T	Charge	Some flexibility.	All flexibility available, all FOs behave like SFOs. Many FlexOffers generated, one each time unit.					
		Switch							
T	1	Charge	Flexibility available for only one time unit. All FOs behave like SFOs.						
		Switch							
	T	Charge	Some flexibility.	Very little flexibility available.	Too much flexibility.	All flexibility available.		Almost all flexibility.	All flexibility available.
		Switch	Very little flexibility.			Some flexibility.	Too much flexibility.	Flexibility varies.	Too much flexibility.

This table describes the various scenarios that happen depending on the parameters. The most interesting ones happen when generation horizon and planning horizon have the same length, and are both greater than one time unit.

Standard FO generation algorithm



Input:
P (parameters)
m (battery model)

Fmin = amount of energy that can be taken
Fmax = amount of energy that can be given

$slices = \emptyset$

P.AType?
P.ASType?

Outer

Inner, Flat

Inner, All At Once

a_{min} = m.P_{min}
a_{max} = m.P_{max}

a_{min} = Fmin/T
a_{max} = Fmax/T

a_{min} = available energy left at time t
a_{max} = chargable energy left at time t

P.PType?

Charge

Discharge

a_{min} = 0

a_{max} = 0

$slices = slices \cup [a_{min}, a_{max}]$
$t = t + 1$

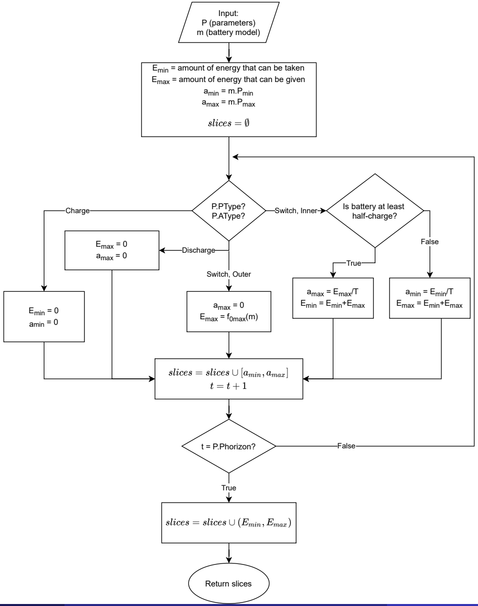
t = P.Phorizon?

False

True

Return slices

TECFO generation algorithm



Input:
P (parameters)
m (battery model)

E_{min} = amount of energy that can be taken
E_{max} = amount of energy that can be given
$a_{min} = m.P_{min}$
$a_{max} = m.P_{max}$
$slices = \emptyset$

P.PType?
P.AType?

Charge

Discharge

Switch, Outer

Switch, Inner

Is battery at least half-charge?

True

False

$E_{max} = 0$
$a_{max} = 0$

$E_{min} = 0$
$a_{min} = 0$

$a_{max} = 0$
$E_{max} = f_{0max}(m)$

$a_{max} = E_{max}/T$
$E_{min} = E_{min}+E_{max}$

$a_{min} = E_{min}/T$
$E_{max} = E_{min}+E_{max}$

$slices = slices \cup [a_{min}, a_{max}]$
$t = t + 1$

t = P.Phorizon?

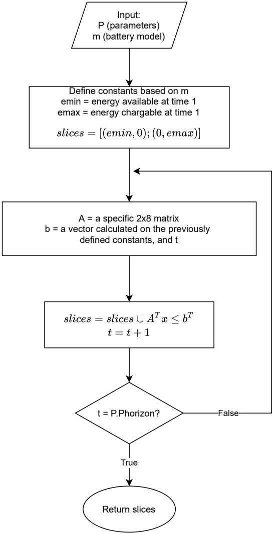
False

True

$slices = slices \cup (E_{min}, E_{max})$

Return slices

DFO generation algorithm



Input:
P (parameters)
m (battery model)

Define constants based on m
emin = energy available at time 1
emax = energy chargable at time 1

$slices = [(emin, 0); (0, emax)]$

A = a specific 2x8 matrix
b = a vector calculated on the previously defined constants, and t

$slices = slices \cup A^T x \leq b^T$
$t = t + 1$

t = P.Phorizon?

Return slices

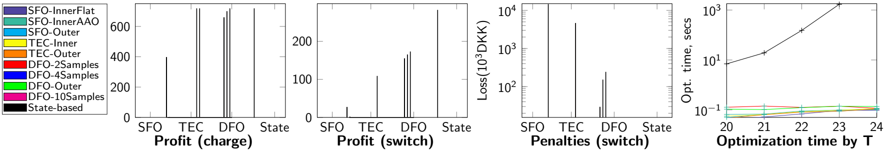
Simulations - use case

For our simulations, we explored a scenario where a prosumer models flexibility for a time horizon of six hours, and tries to maximize profit by buying energy when the price is low, and selling it when the price is high. Data for spot and imbalance prices have been taken from a sample in NordPool.

We compared the results obtained with the FO models against a baseline given by an LTI exact state-space model. The comparison metric is economic revenue, defined as the highest profit that can be obtained by scheduling energy consumption according to the flexibility model's constraints.

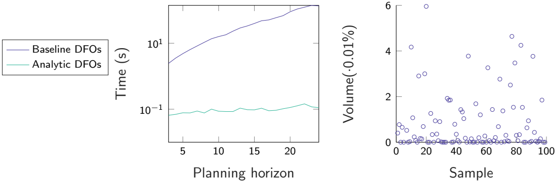
Results

Optimization of a single battery



Profit (charge) **Profit (switch)** **Penalties (switch)** **Optimization time by T**

Analytic outer DFO generation

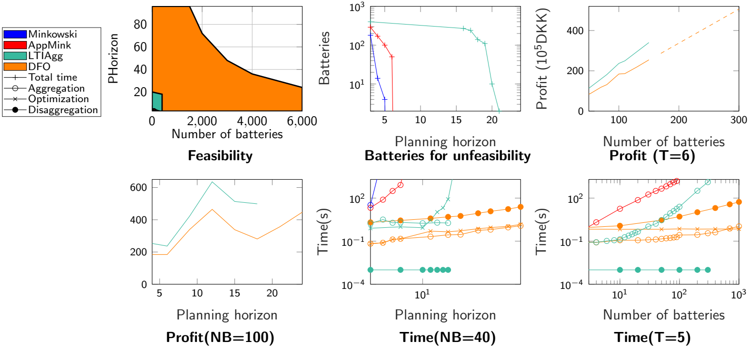


Analytic FO generation time **Volume difference**

Results

Aggregation

We compare our results against three baselines: aggregation by Minkowski sum, aggregation by an approximate Minkowski sum, and **LTI Aggregation**, which is a single LTI state-space model that describes the functioning of all the aggregated batteries.



Feasibility

Batteries for unfeasibility

Profit (T=6)

Profit(NB=100)

Time(NB=40)

Time(T=5)

Conclusion

Our motivation was to find a representation for battery flexibility which is scalable with respect to optimization for long time horizons, and easy to aggregate with other flexible loads.

We proposed the FlexOffer model as a solution, and showed that it complies to those requirements. We showed generation algorithms, included an analytic one for outer DFOs, and we defined a new exact aggregation technique which has used as a baseline.

We plan to extend the FO concept for capturing more flexibility from complex processes (like *switching*), extend it to more device types, and consider more flexibility metrics.

Thanks

Thank you for your attention!