How to Model Competing Flexibility Options (Fast)

Christoph Schimeczek[1], Felix Nitsch[2], Johannes Kochems[1], Kristina Nienhaus[1]

31st March 2026

[1]German Aerospace Center (DLR), Institute of Networked Energy Systems, Stuttgart
[2]BOKU University, Institute of Sustainable Economic Development, Vienna

Welcome

Image source: DLR e.V.

Motivation

Flexibility modelling with AMIRIS

Transformation to **renewable-dominated** energy system

> Rising shares of fluctuating renewable energies
> Merit-order effect: falling market values of RES
> Refinancing uncertain

More **flexibility options** are installed

> New flexibility technologies emerge
> Increasing competition: falling margins
> Refinancing uncertain

Aim

- Understand **market effects** of renewables and flexibilities
- Consider **actors' behaviour**, uncertainty and market distortions caused by regulatory framework
- Study **policy instruments** to incite system-friendly investment and operational decisions

AMIRIS: Overview

Image source: DLR e.V.

AMIRIS

Agent-Based Market Model for the Investigation of Renewable and Integrated Energy Systems

- **Agent-based** model for power markets
- Models **business-oriented** dispatch decisions under different regulatory framework conditions
- Focus on **renewable** energy sources and **flexibility** options
- Developed **open source** without copyleft

https://wonderl.ink/@amiris

AMIRIS

Agent-Based Market Model for the Investigation of Renewable and Integrated Energy Systems

- Simulates **trading** of supply and demand
- Considers **uncertainty** and market distortions
- Resolution: hourly (temporal) – market zones (spatial)
- Runs yearly simulations on laptops in less than a minute

https://wonderl.ink/@amiris

AMIRIS
Input & Output

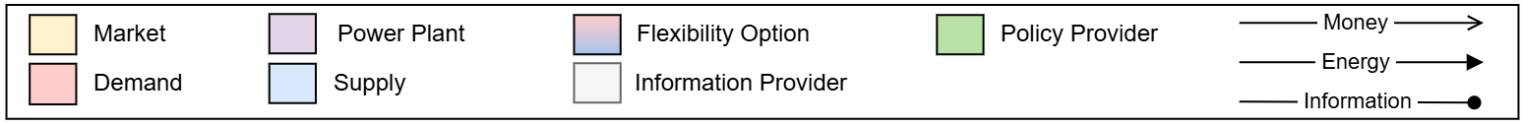
Input

- Power plant park
  - Efficiencies
  - Outages
  - Feed-in potential
- Demand
- Fuel prices
- CO_2 prices

Output

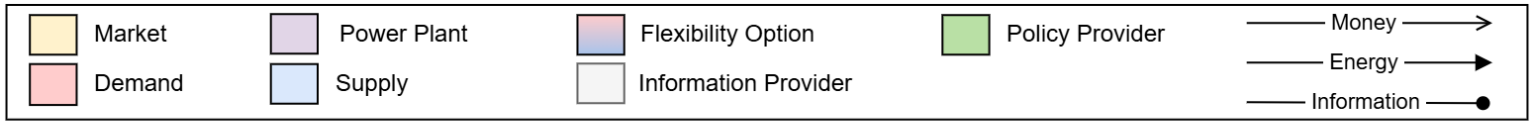
- Electricity prices
- Power plant dispatch
- Market values
- CO_2 emissions
- System costs
- Costs for support instruments

AMIRIS
Agent types



AMIRIS
Agent types

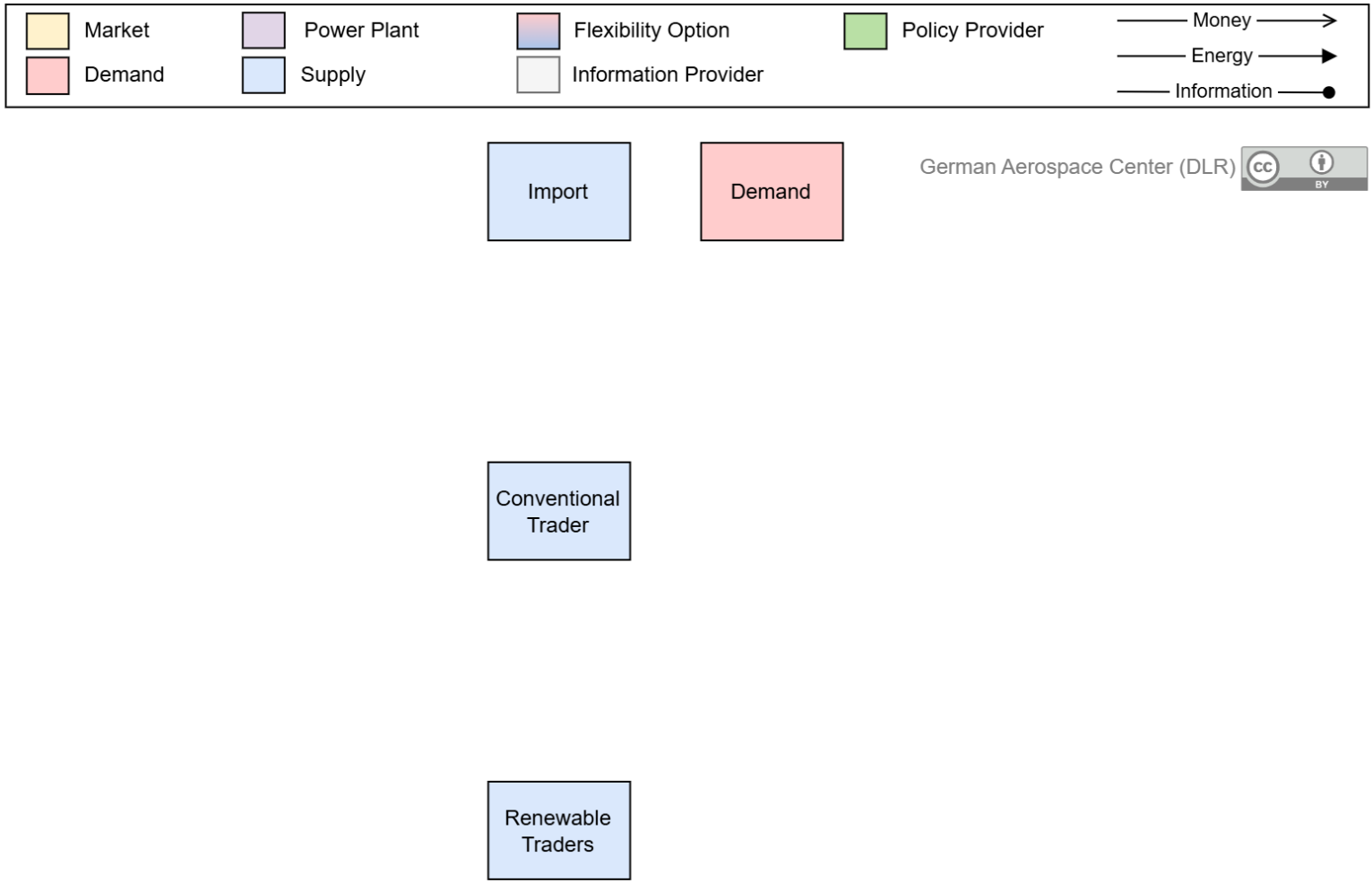
Markets
- Determine prices



AMIRIS
Agent types

Markets
- Determine prices

Traders
- Fulfil marketing strategies

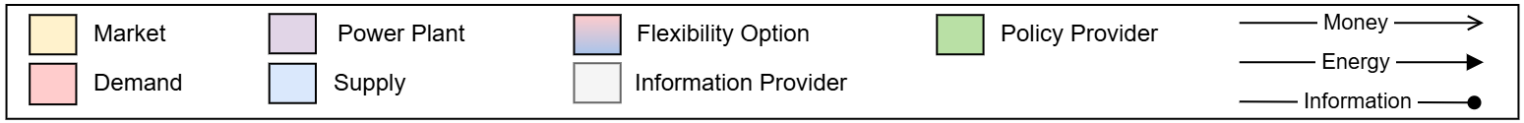


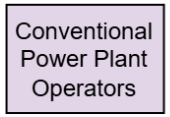
AMIRIS
Agent types

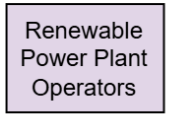
Markets
- Determine prices

Traders
- Fulfil marketing strategies

Plant operators
- Control power plants







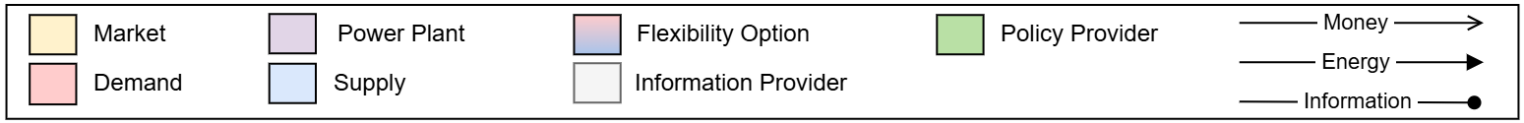
AMIRIS
Agent types

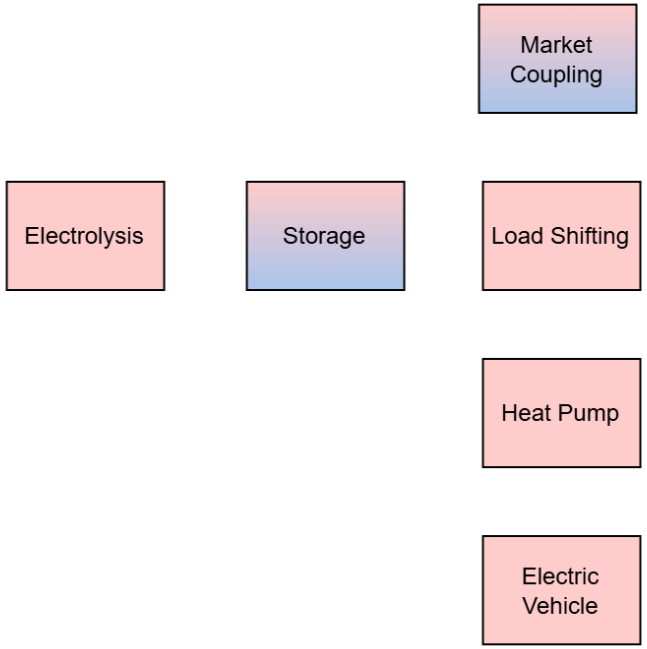
Markets
- Determine prices

Traders
- Fulfil marketing strategies

Plant operators
- Control power plants

Flexibility providers
- Optimise dispatch





AMIRIS
Agent types

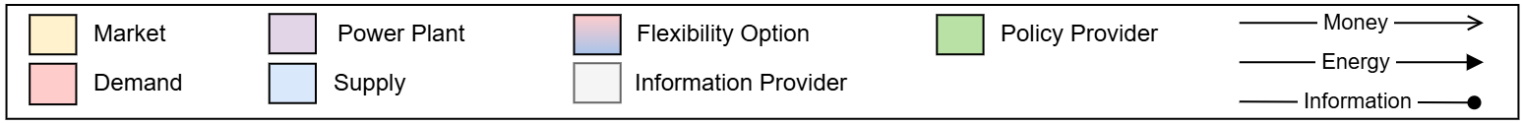
Markets
- Determine prices

Traders
- Fulfil marketing strategies

Plant operators
- Control power plants

Flexibility providers
- Optimise dispatch

Information provider
- Create forecasts



AMIRIS
Agent types

Markets
- Determine prices

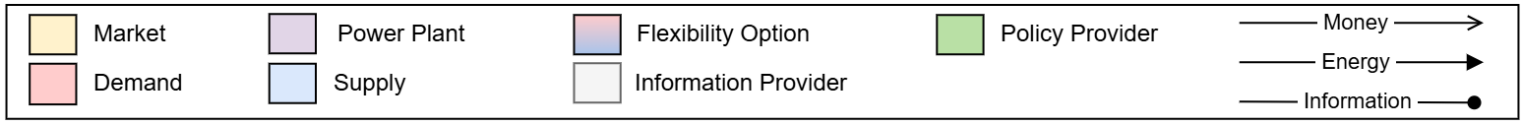
Traders
- Fulfil marketing strategies

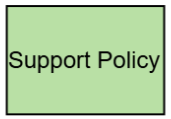
Plant operators
- Control power plants

Flexibility
- Optimise dispatch

Information provider
- Create forecasts

Policy
- Provide support





AMIRIS
Agent types

Markets
- Determine prices

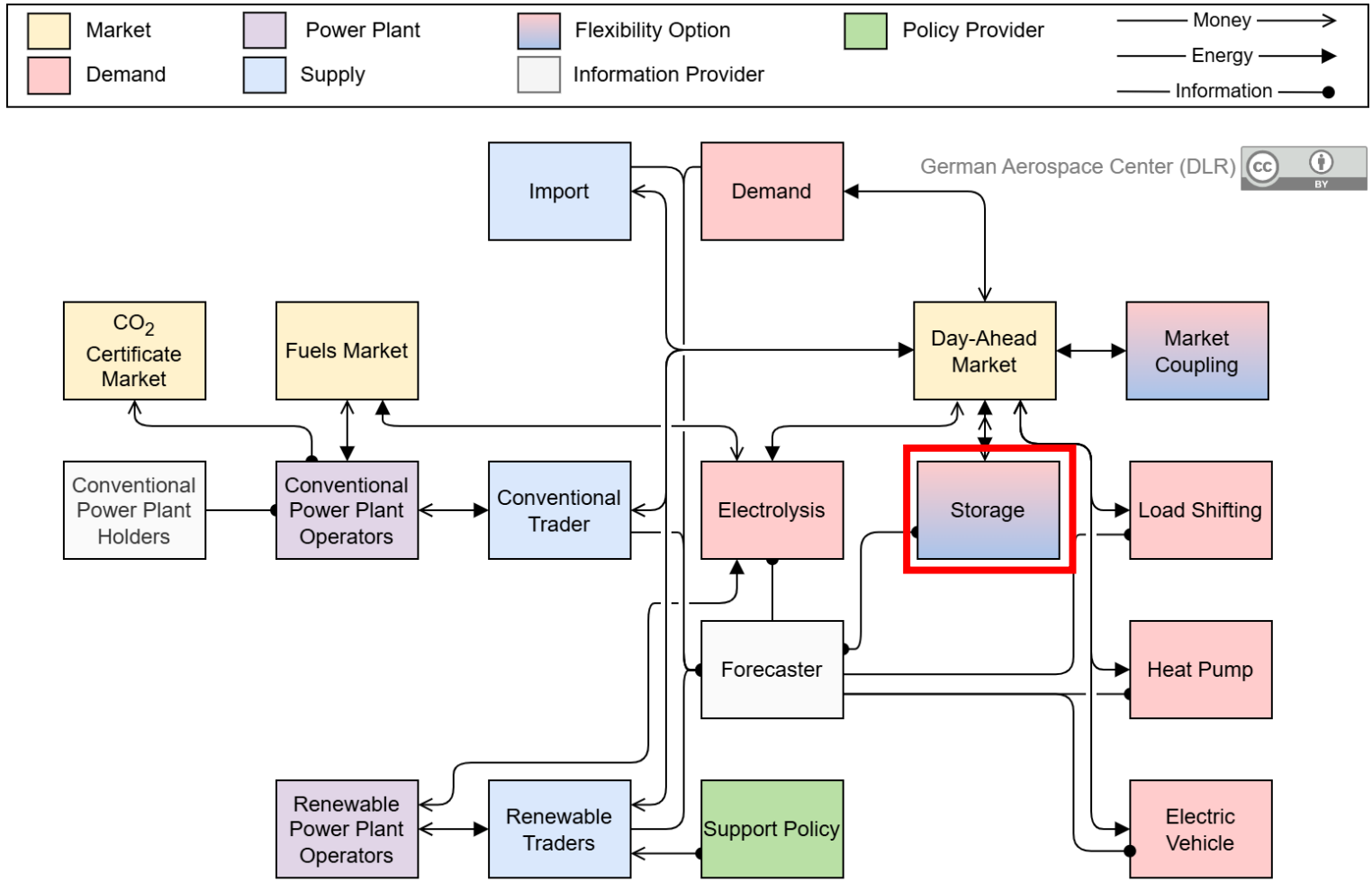
Traders
- Fulfil marketing strategies

Plant operators
- Control power plants

Flexibility providers
- Optimise dispatch

Information provider
- Create forecasts

Policy
- Provide support



Modelling Competing Flexibility

Image source: DLR e.V.

Modelling Competing Flexibility

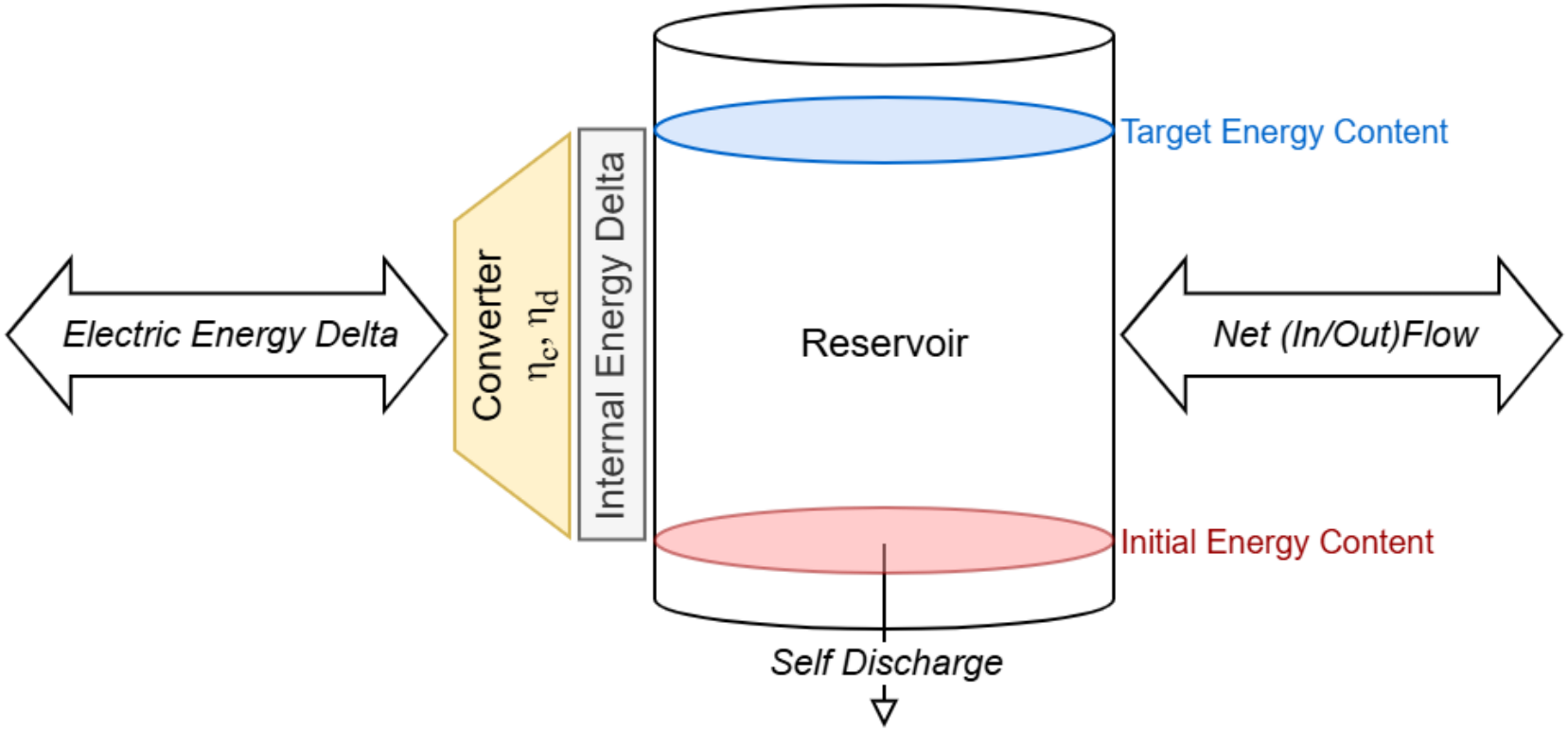
The Flexibility Device

Reservoir
- Storage capacity in MWh

Converter
- (Dis-) charging power in MW
- (Dis-) charging efficiency

Transitions
- Initial → target energy content
- Internal energy delta → external electric energy delta
- Self discharge
- Inflows / outflows

Modelling Capabilities
- Pumped hydro (+inflow)
- Hydro reservoir
- Battery storage
- Sector coupling tech (simplified) (Heat pumps, EVs, load shifting,..)



Modelling Competing Flexibility

Device Configuration

```
- Type: GenericFlexibilityTrader # Pumped Hydro
  Id: 7
  Attributes:
    Device:
      GrossChargingPowerInMW: 8988.8
      NetDischargingPowerInMW: 7120.
      ChargingEfficiency: 0.89
      DischargingEfficiency: 0.89
      EnergyContentUpperLimitInMWH: 40000.
      InitialEnergyContentInMWH: 1000.
      NetInflowPowerInMW: 0
      SelfDischargeRatePerHour: 0
```

At start of simulation

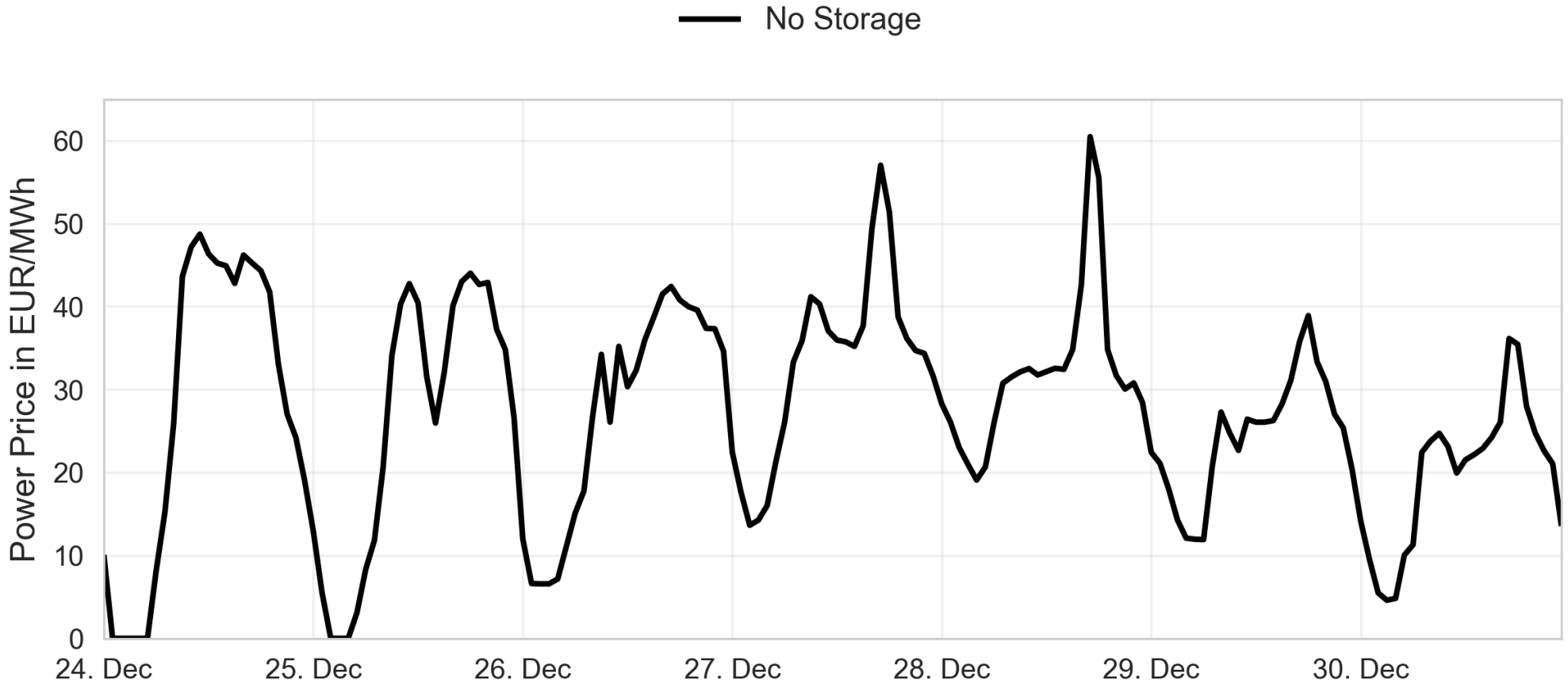
Makes things slow if != 0

Modelling Competing Flexibility

The Task

When to charge or to discharge…

→ Use electricity price *forecast* → Optimise dispatch with *dynamic programming*

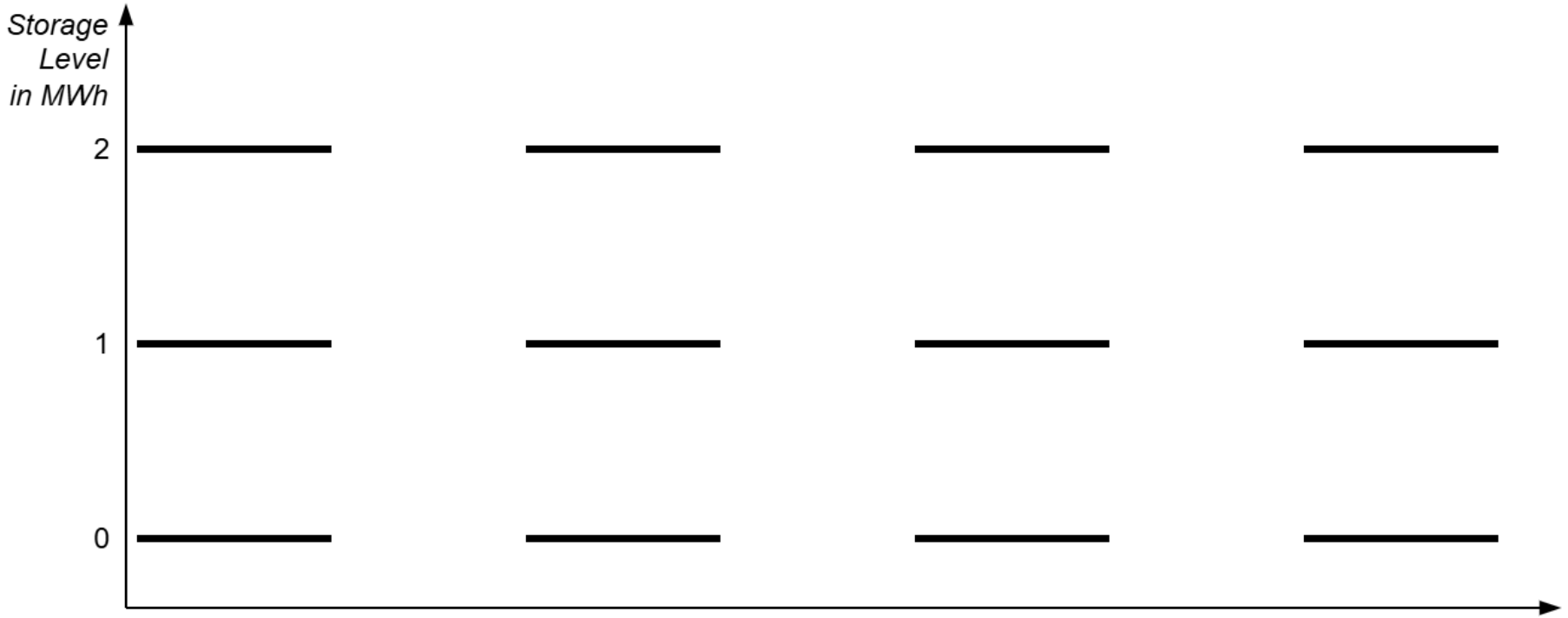


Modelling Competing Flexibility

Dynamic Programming

Discretise storage content

- e.g. 10 MWh steps



Modelling Competing Flexibility

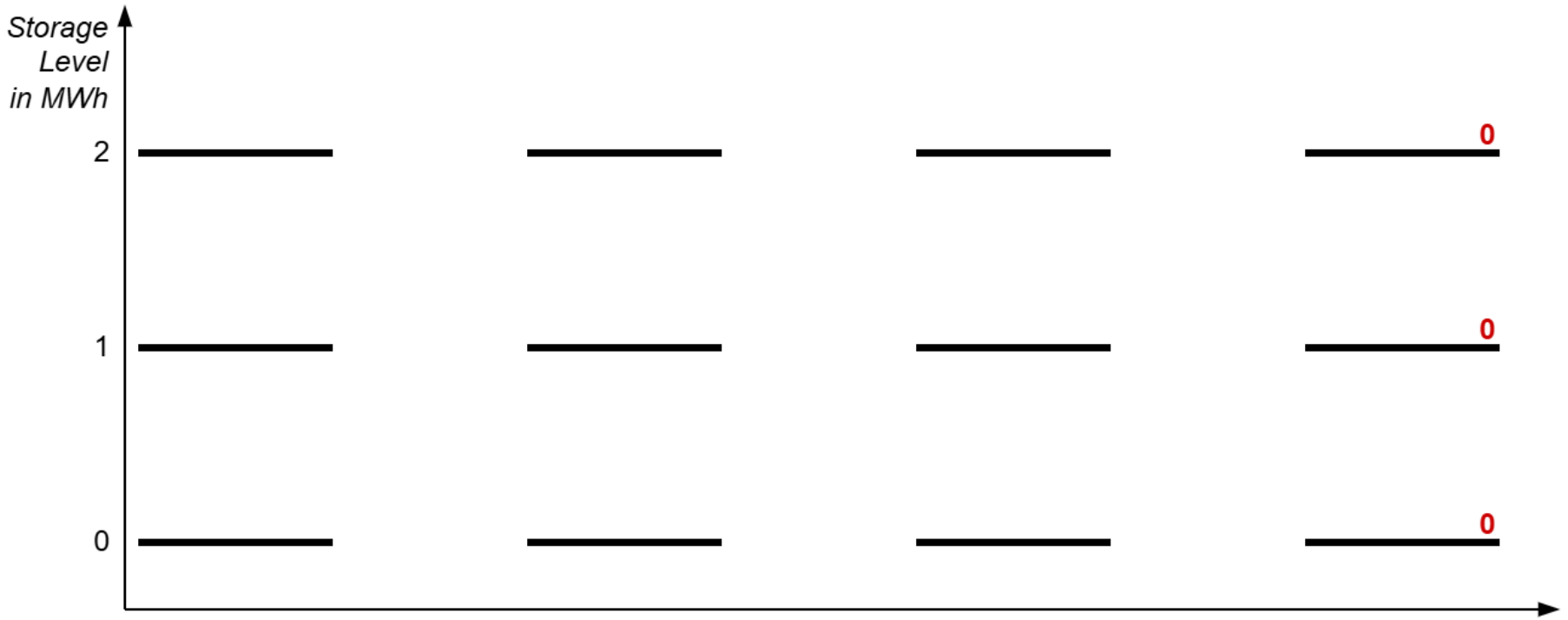
Dynamic Programming

Discretise storage content

- e.g. 10 MWh steps

Assign final value

- e.g. zero, or water value



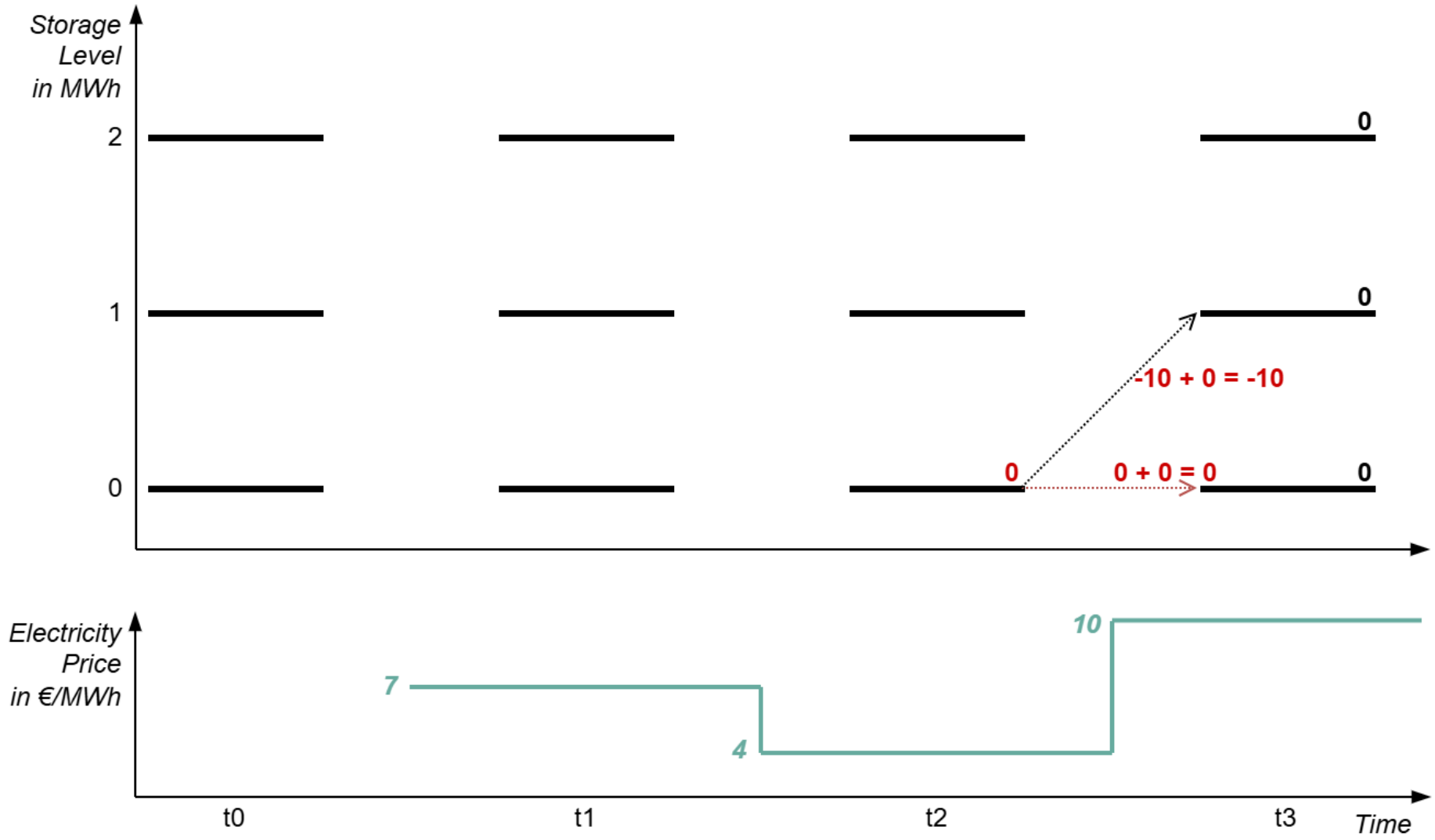
Modelling Competing Flexibility

Dynamic Programming

Discretise storage content
- e.g. 10 MWh steps

Assign final value
- e.g. zero, or water value

Evaluate transitions
- Pick optimisation target
- Backwards in time
- Assess all allowed transitions
- Choose best transition



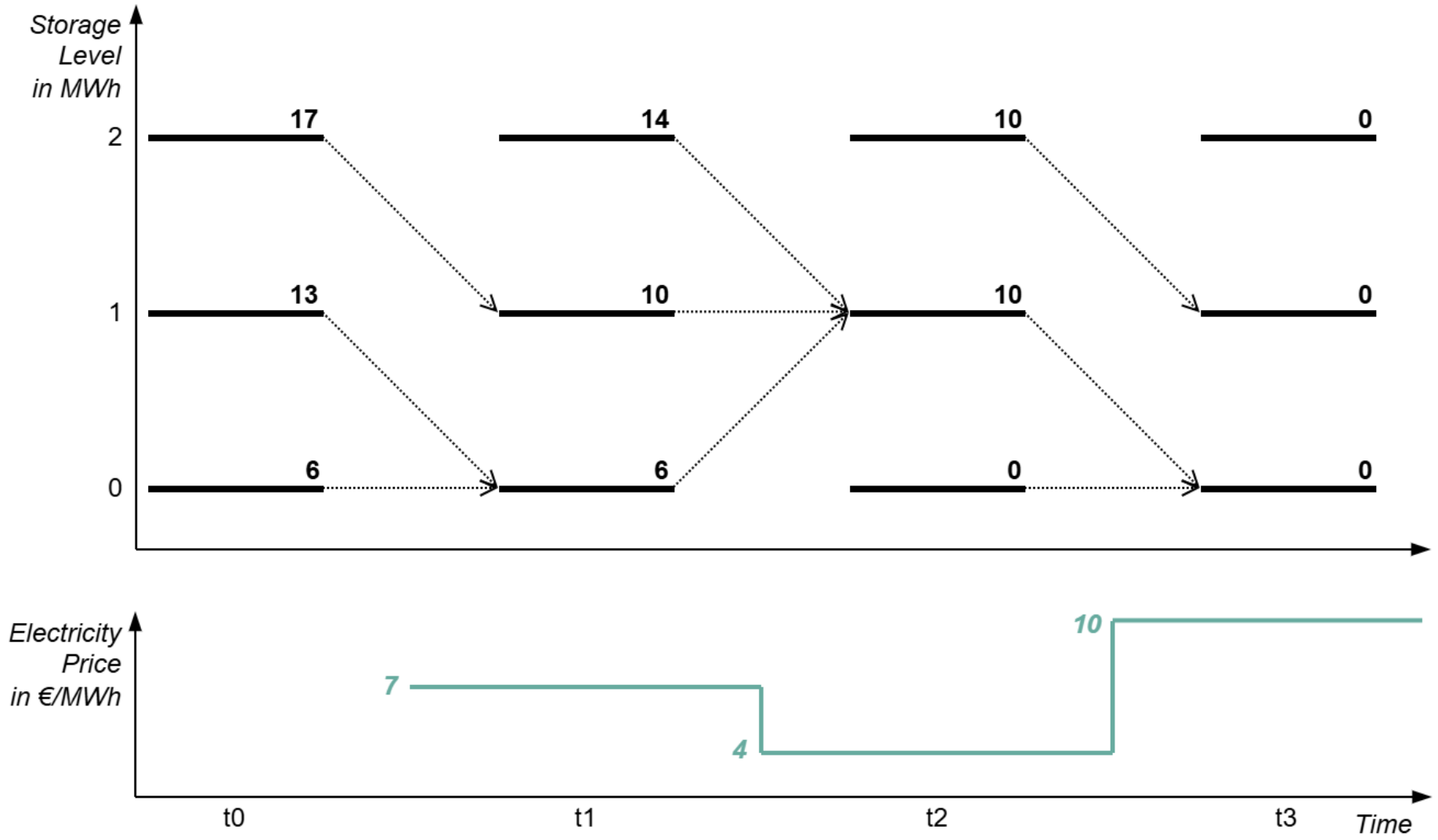
Modelling Competing Flexibility

Dynamic Programming

Discretise storage content
- e.g. 10 MWh steps

Assign final value
- e.g. zero, or water value

Evaluate transitions
- Pick optimisation target
- Backwards in time
- Assess all allowed transitions
- Choose best transition



Modelling Competing Flexibility

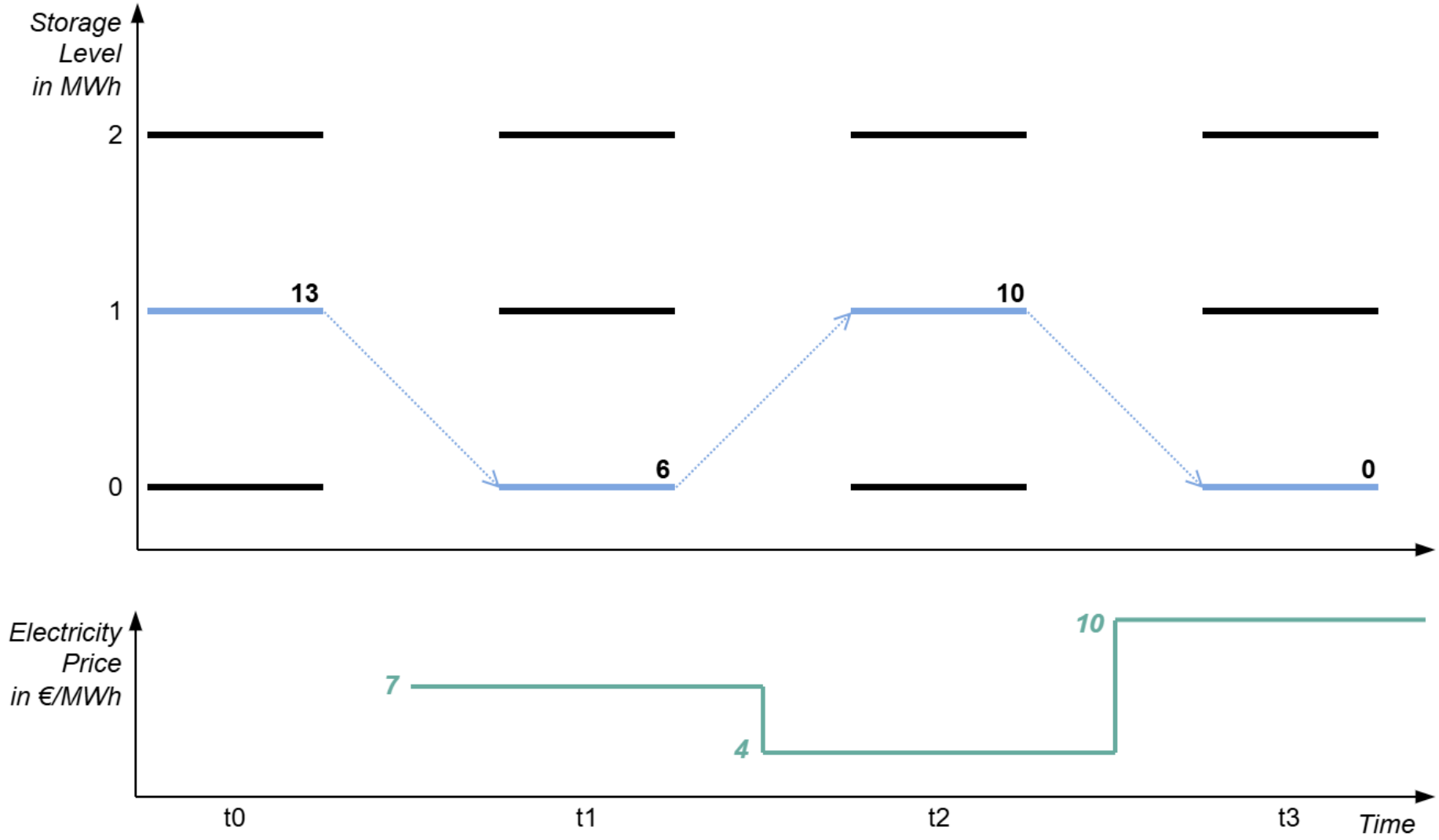
Dynamic Programming

Discretise storage content
- e.g. 10 MWh steps

Assign final value
- e.g. zero, or water value

Evaluate transitions
- Pick optimisation target
- Backwards in time
- Assess all allowed transitions
- Choose best transition

Plan dispatch
- Begin at current state
- Follow best transitions



Modelling Competing Flexibility

Dynamic Programming

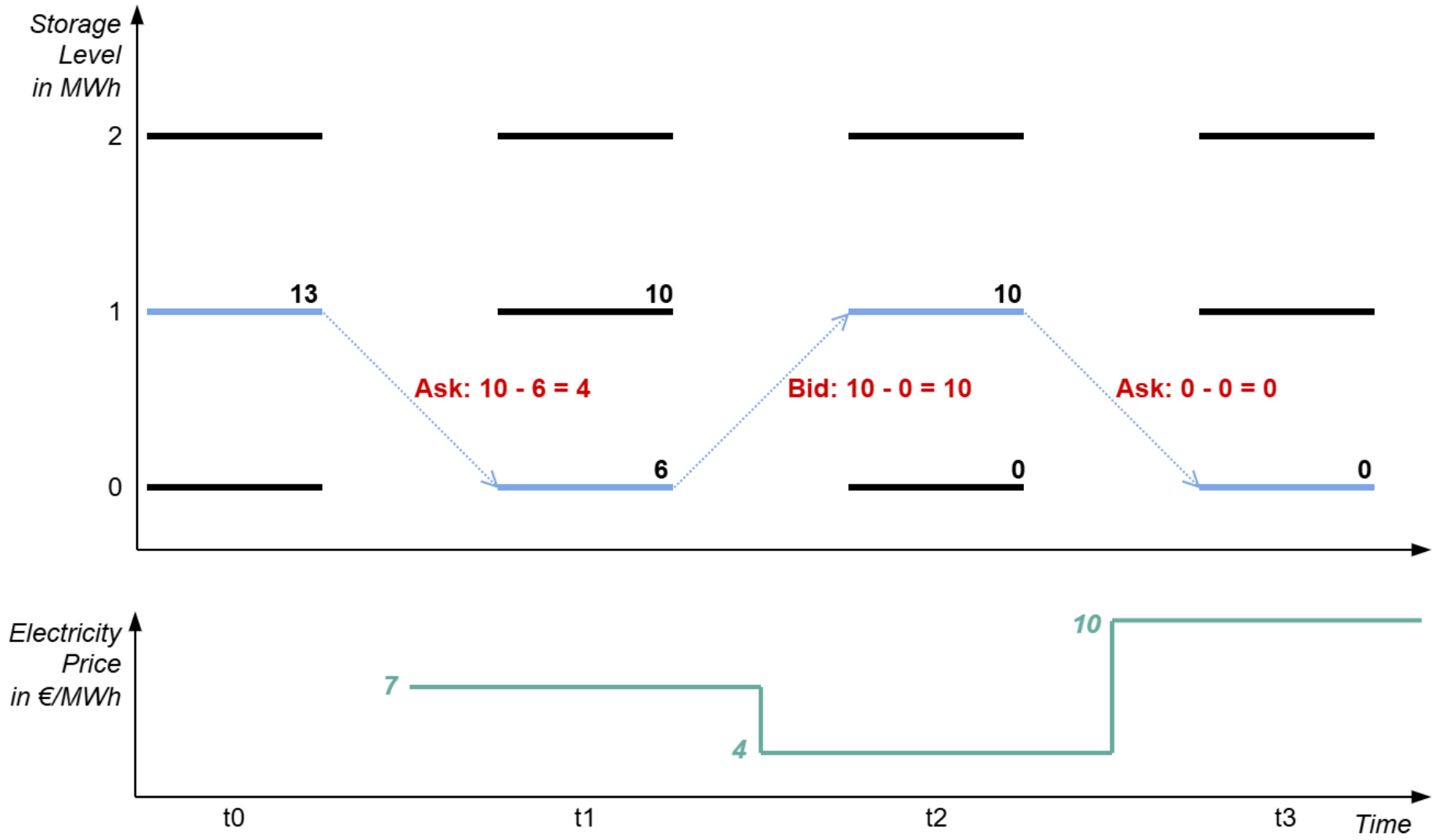
Discretise storage content
- e.g. 10 MWh steps

Assign final value
- e.g. zero, or water value

Evaluate transitions
- Pick optimisation target
- Backwards in time
- Assess all allowed transitions
- Choose best transition

Plan dispatch
- Begin at current state
- Follow best transitions

Bidding strategy
- Consider expected value



Modelling Competing Flexibility

Strategic Configuration

```
Assessment:
   Type: MIN_SYSTEM_COST
StateDiscretisation:
   Type: STATE_OF_CHARGE
   PlanningHorizonInHours: 168
   EnergyResolutionInMWH: 160
Bidding:
   Type: ENSURE_DISPATCH
   SchedulingHorizonInHours: 24
```

Optimisation target; alternative: MAX_PROFIT

Energy states only (load shifting also considers time)

Foresight during planning

Granularity of discretisation

Bid with max / min price; alternative STORAGE_CONTENT_VALUE

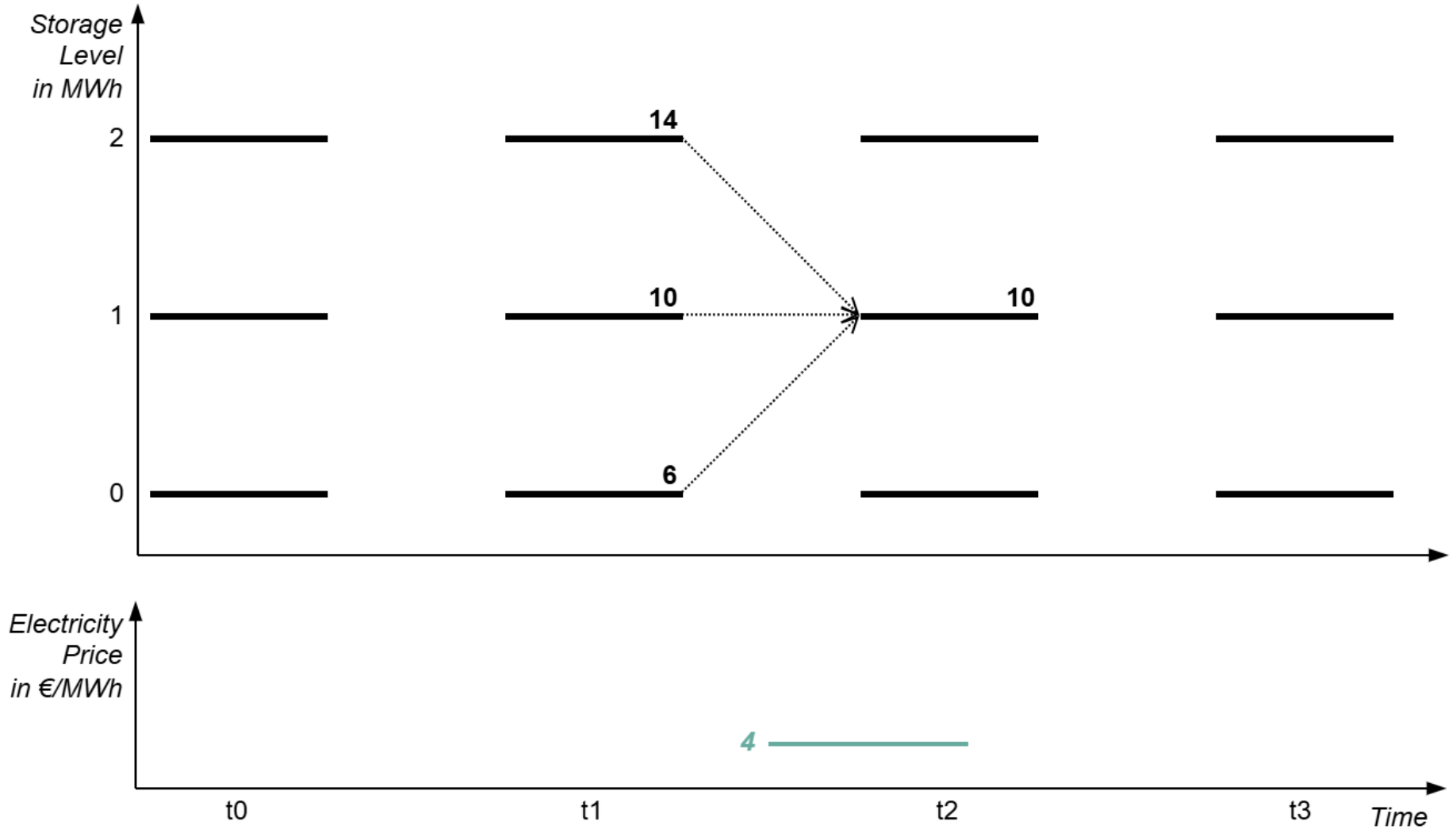
Planned re-scheduling (rolling horizon)

Modelling Competing Flexibility

Price impacts

Static price assumption

- Is wrong



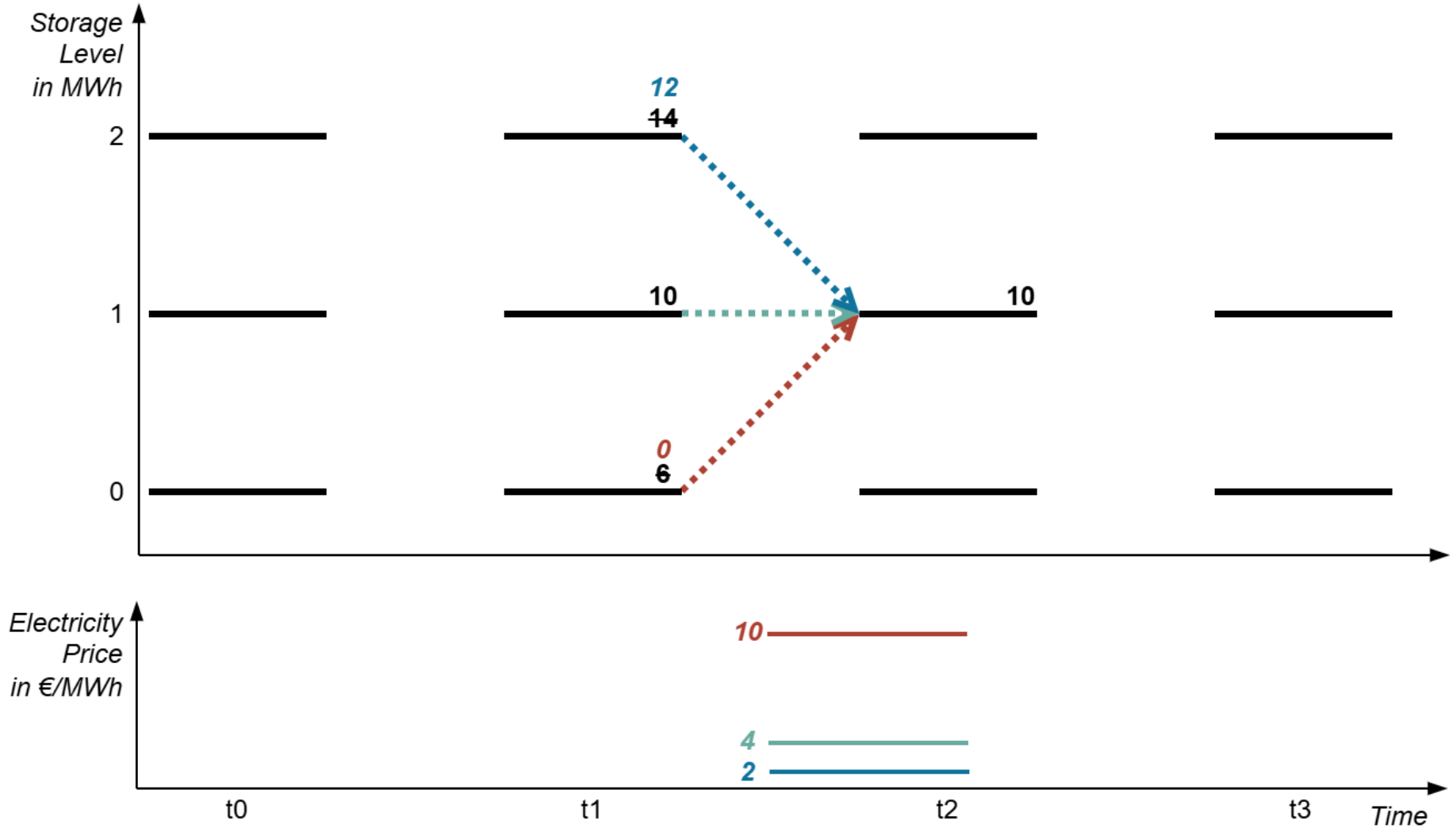
Modelling Competing Flexibility

Price impacts

Static price assumption

- Is wrong

Prices depend on dispatch



Modelling Competing Flexibility

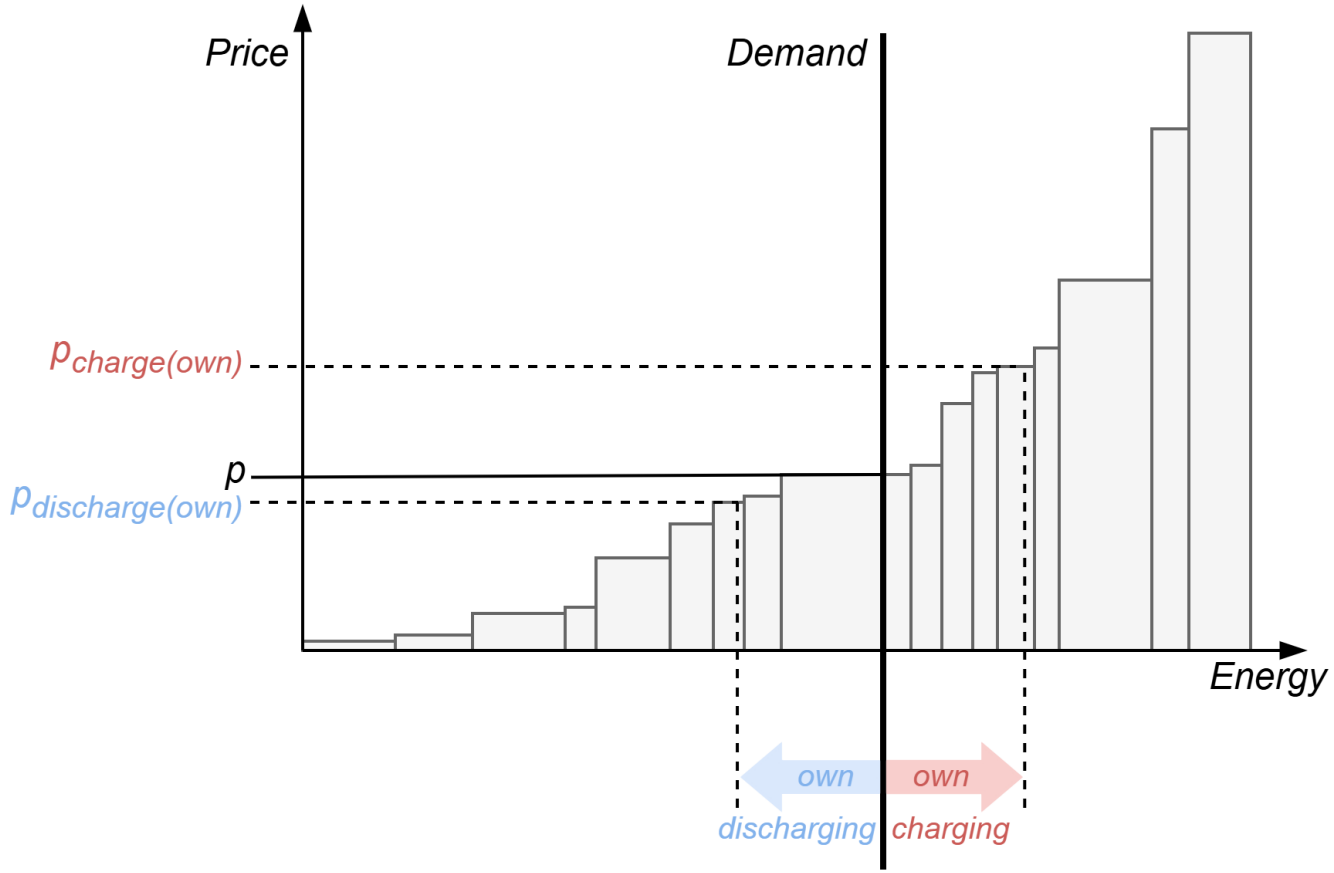
Price impacts

Static price assumption

- Is wrong

Prices depend on dispatch

→ Merit order forecast

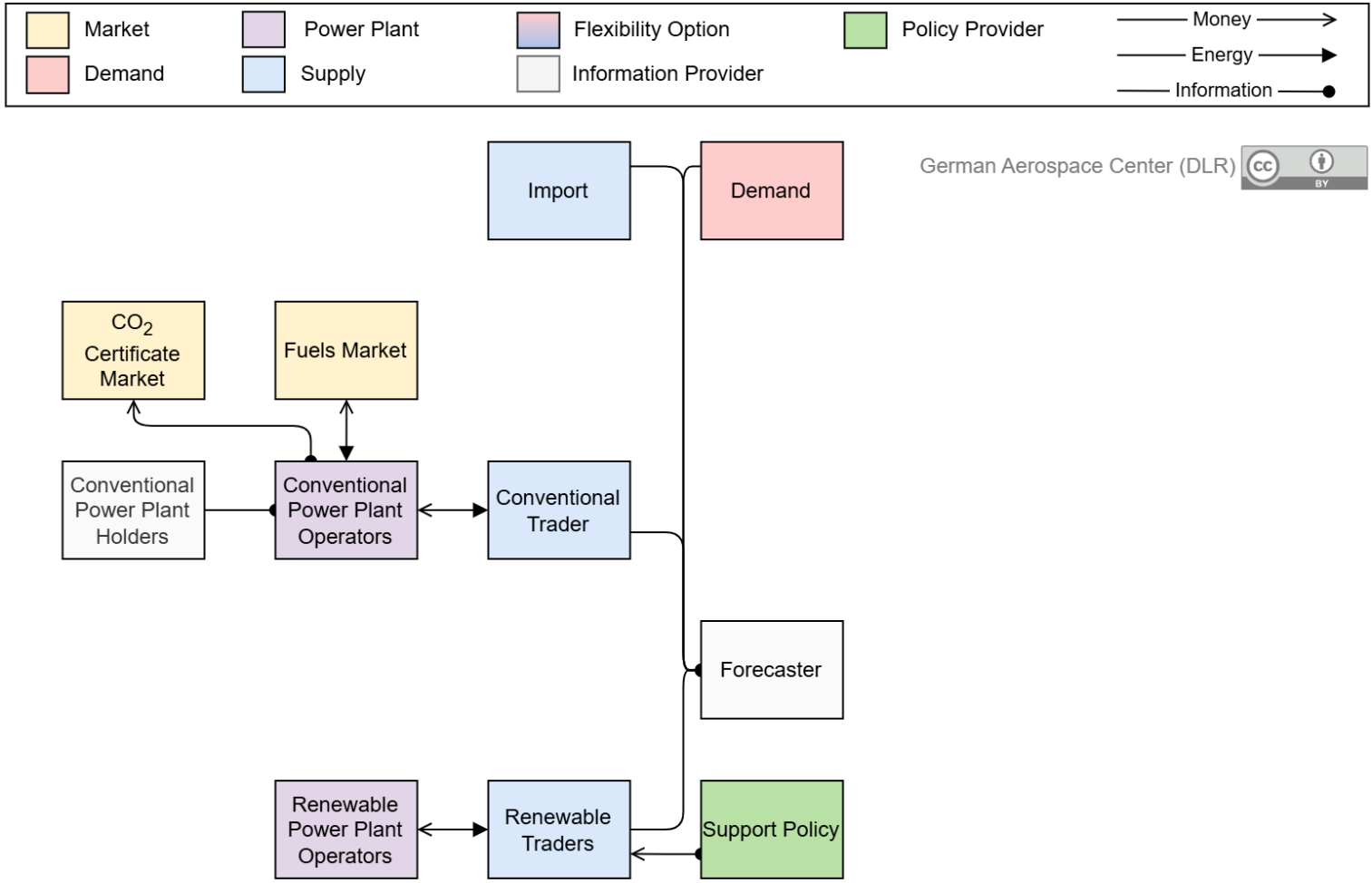


Modelling Competing Flexibility

Forecast: Inflexible Units

Inflexible units

- Provide their bid forecasts



Modelling Competing Flexibility

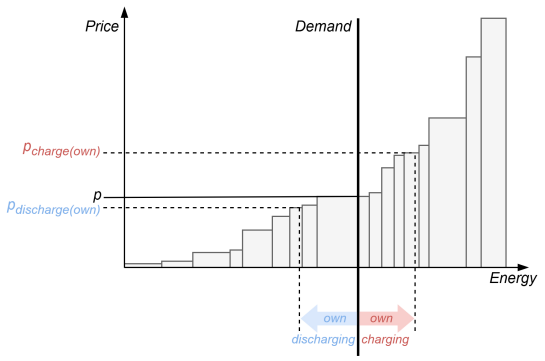
Forecast: Inflexible Units

Inflexible units

- Provide their bid forecasts

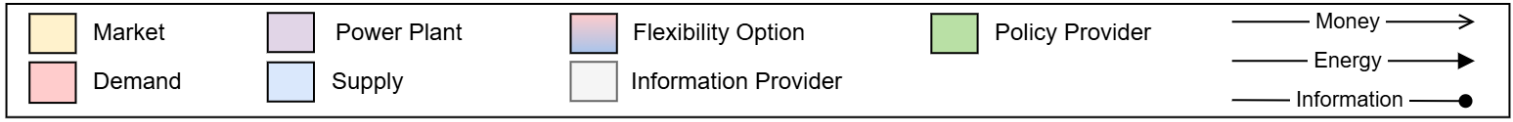
Forecaster

- Derives merit order



Modelling Competing Flexibility

Forecast: Inflexible Units



Inflexible units

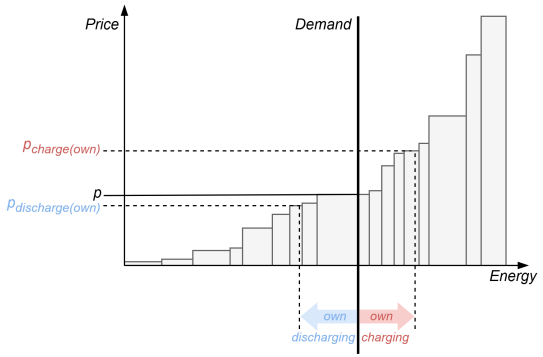
- Provide their bid forecasts

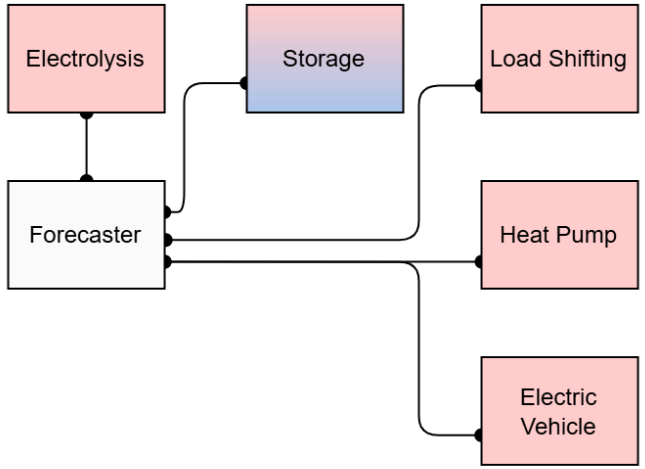
Forecaster

- Derives merit order
- Sends effective prices @ dispatched MWh

Flexibilities

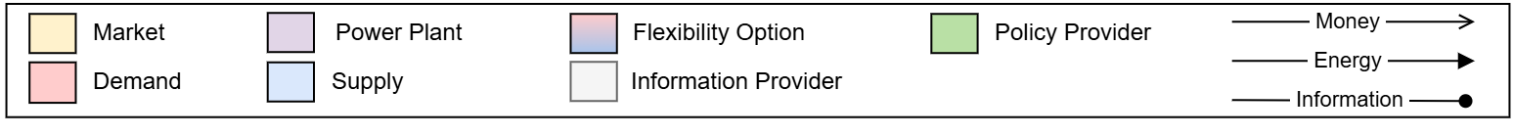
- Consider price changes in their dispatch optimisation





Modelling Competing Flexibility

Forecast: Inflexible Units



Inflexible units

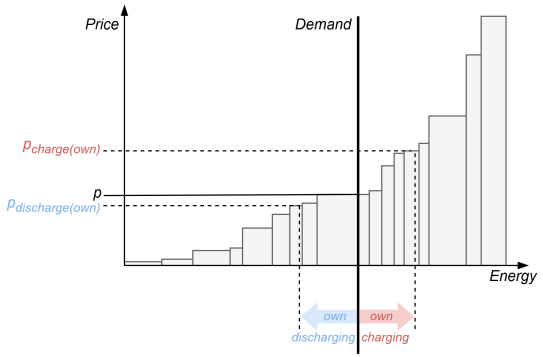
- Provide their bid forecasts

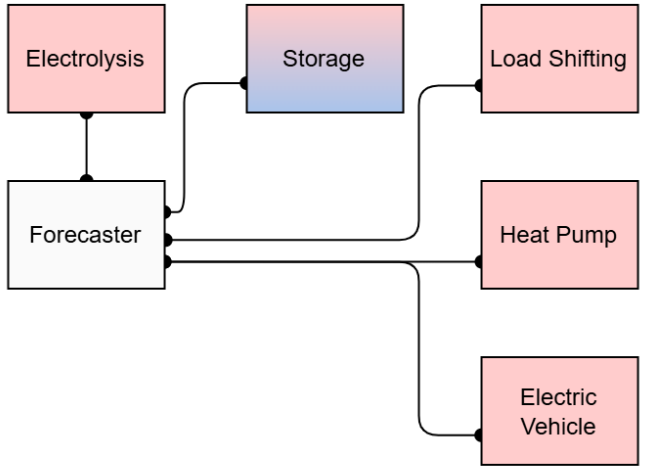
Forecaster

- Derives merit order
- Sends effective prices @ dispatched MWh

Flexibilities

- Consider price changes in their dispatch optimisation

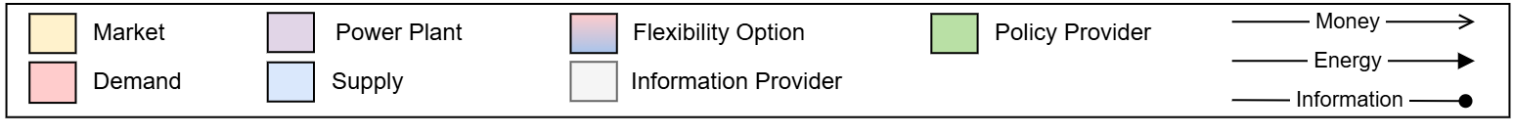




But how consider competition?

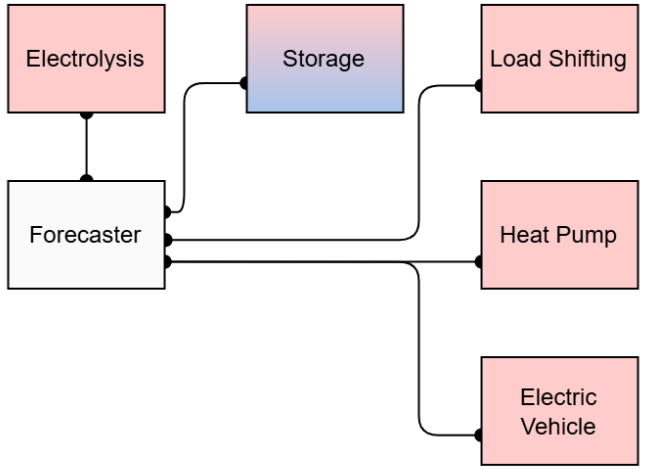
Modelling Competing Flexibility

Forecast: Flexibility Units



Flexibilities

- Provide their historic dispatch



Modelling Competing Flexibility

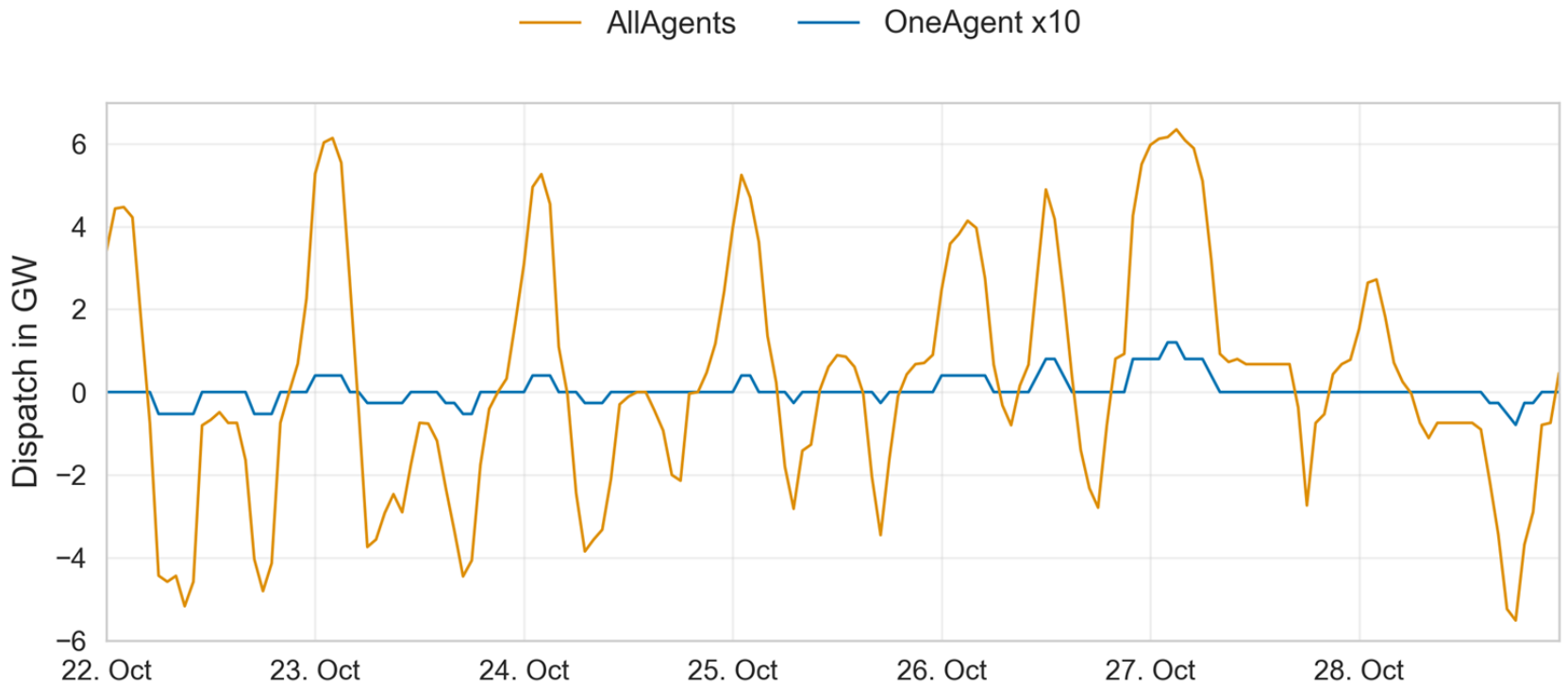
Forecast: Flexibility Units

Flexibilities

- Provide their historic dispatch

Forecaster

- Compares each unit's dispatch with total dispatch



Modelling Competing Flexibility

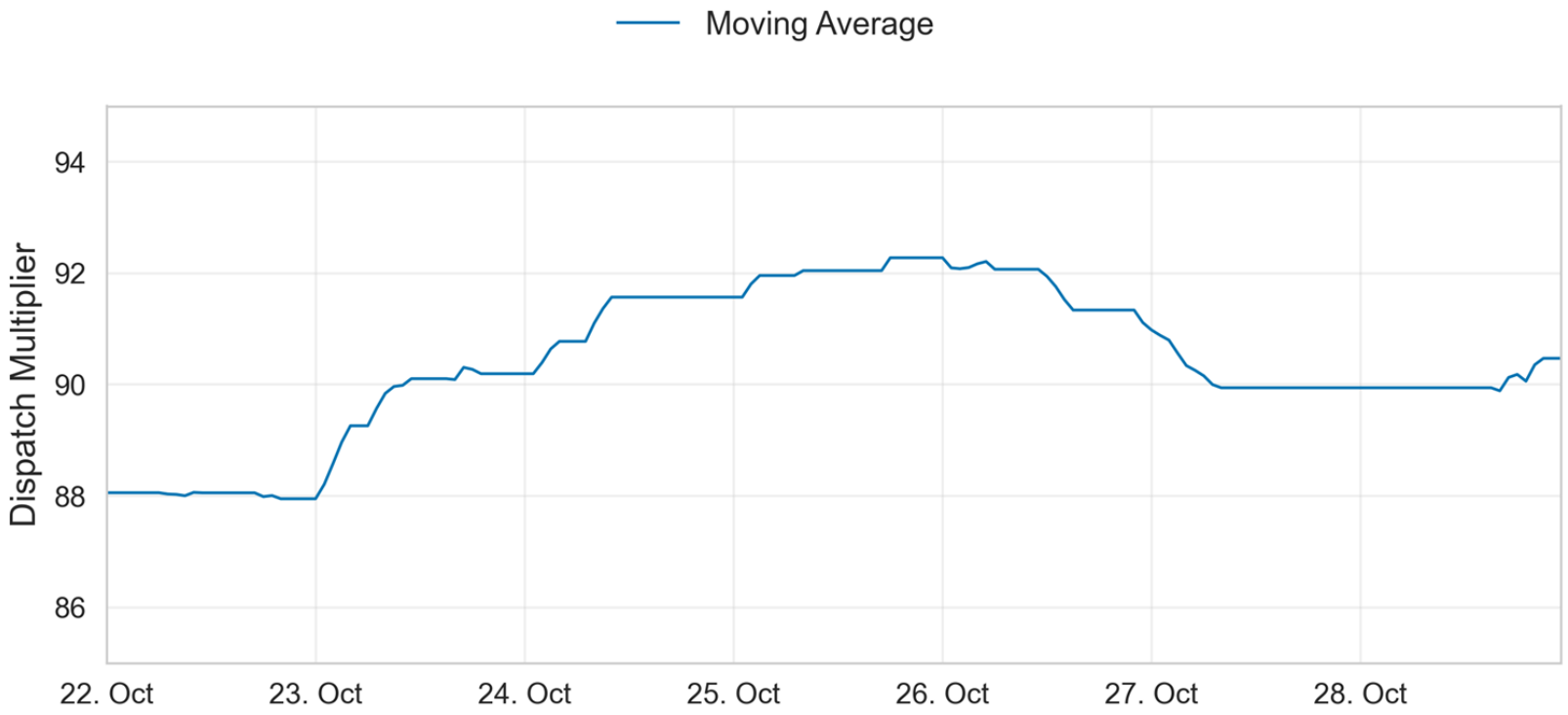
Forecast: Flexibility Units

Flexibilities

- Provide their historic dispatch

Forecaster

- Compares each unit's dispatch with total dispatch
- Derives an average "dispatch multiplier"



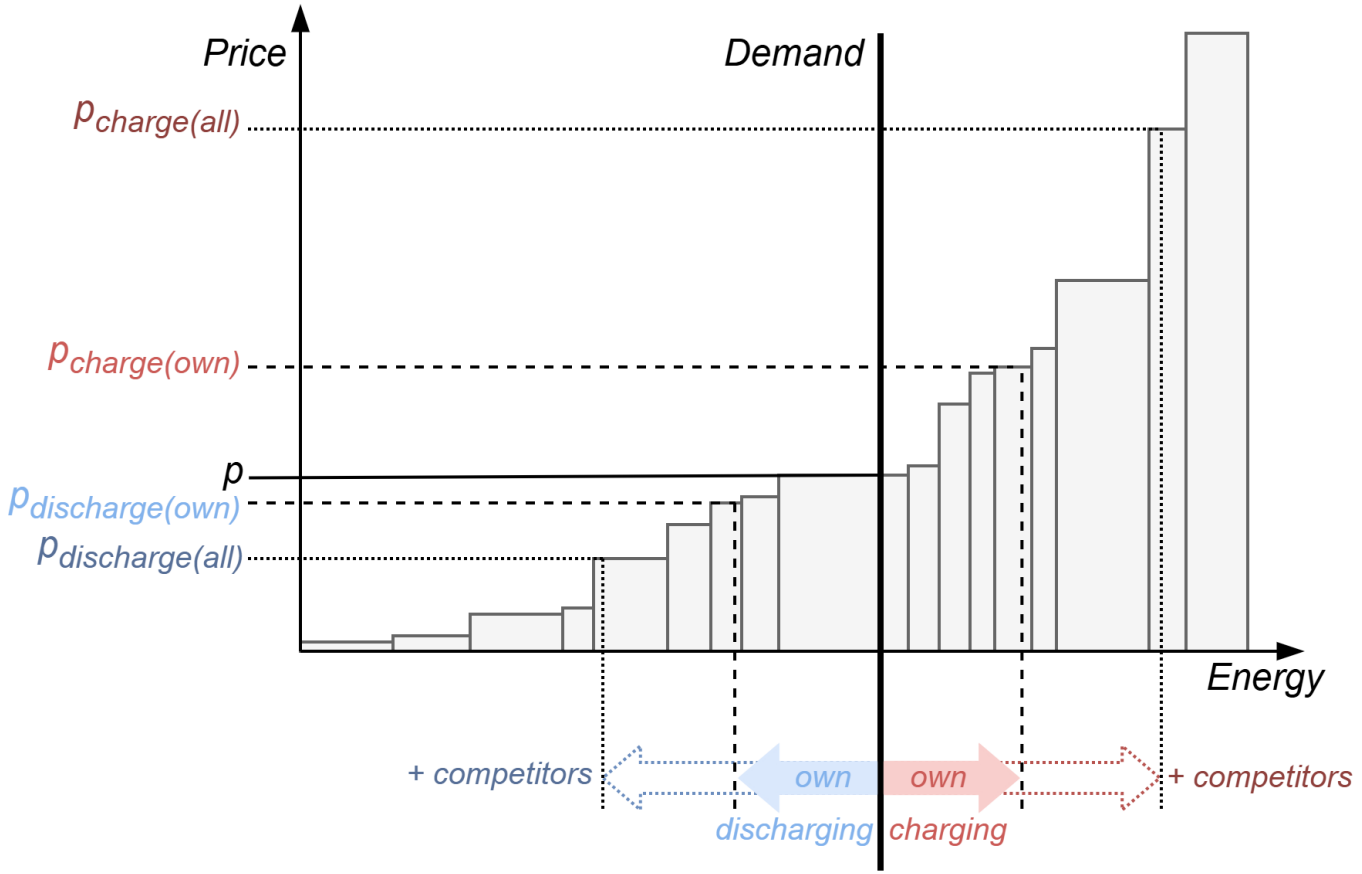
Modelling Competing Flexibility

Forecast: Flexibility Units

Flexibilities
- Provide their historic dispatch

Forecaster
- Compares each unit's dispatch with total dispatch
- Derives an average "dispatch multiplier"
- Sends effective prices @ dispatched MWh

Flexibilities
- Consider price changes in their dispatch optimisation using the multiplier



Modelling Competing Flexibility

Dispatch Multiplier Estimation

At the beginning

- No dispatch history exists
- → use average share converter power $C_k^{\mathrm{c}} + C_k^{\mathrm{d}}$ to derive initial multiplier $m_{k,0}$ of unit k

$$m_{k,0} = \frac{\sum_{j=1}^{N}(C_j^{\mathrm{c}} + C_j^{\mathrm{d}})}{C_k^{\mathrm{c}} + C_k^{\mathrm{d}}}$$

Update estimate after each market clearing

- Lower importance of current multiplier estimate by decay factor α derived from decay time τ
- Calculate unit k's multiplier $m_{k,t}$ at time t as inverse share on total net awarded energy A

$$\alpha = \exp\left(-\frac{1}{\tau}\right) \qquad \alpha^{\tau} = \frac{1}{\mathrm{e}}$$

$$m_{k,t} = \frac{\sum_{j=1}^{N} A_{j,t}}{A_{k,t}}$$

- Derive new moving average multiplier $\bar{m}_{k,t}$
- Older values' importance decays

$$\bar{m}_{k,t} = \frac{\sum_{t'=0}^{t} m_{k,t} \cdot \alpha^{t-t'}}{\sum_{t'=0}^{t} \alpha^{t-t'}}$$

Stabilisation at beginning

- Increase weight of initial estimate by ω_0

$$\bar{m}_{k,t} = \frac{m_{k,0} \cdot \omega_0 \cdot \alpha_t + \sum_{t'=1}^{t} m_{k,t} \cdot \alpha^{t-t'}}{\omega_0 \cdot \alpha_t + \sum_{t'=1}^{t} \alpha^{t-t'}}$$

Modelling Competing Flexibility

Forecast Configuration

```
- Type: SensitivityForecaster
  Id: 6
  Attributes:
    Clearing: *clearingParameters
    ForecastPeriodInHours: 730
    MultiplierEstimation:
      InitialEstimateWeight: 6
      DecayInterval: 168
      IgnoreAwardFactor: 1000
```

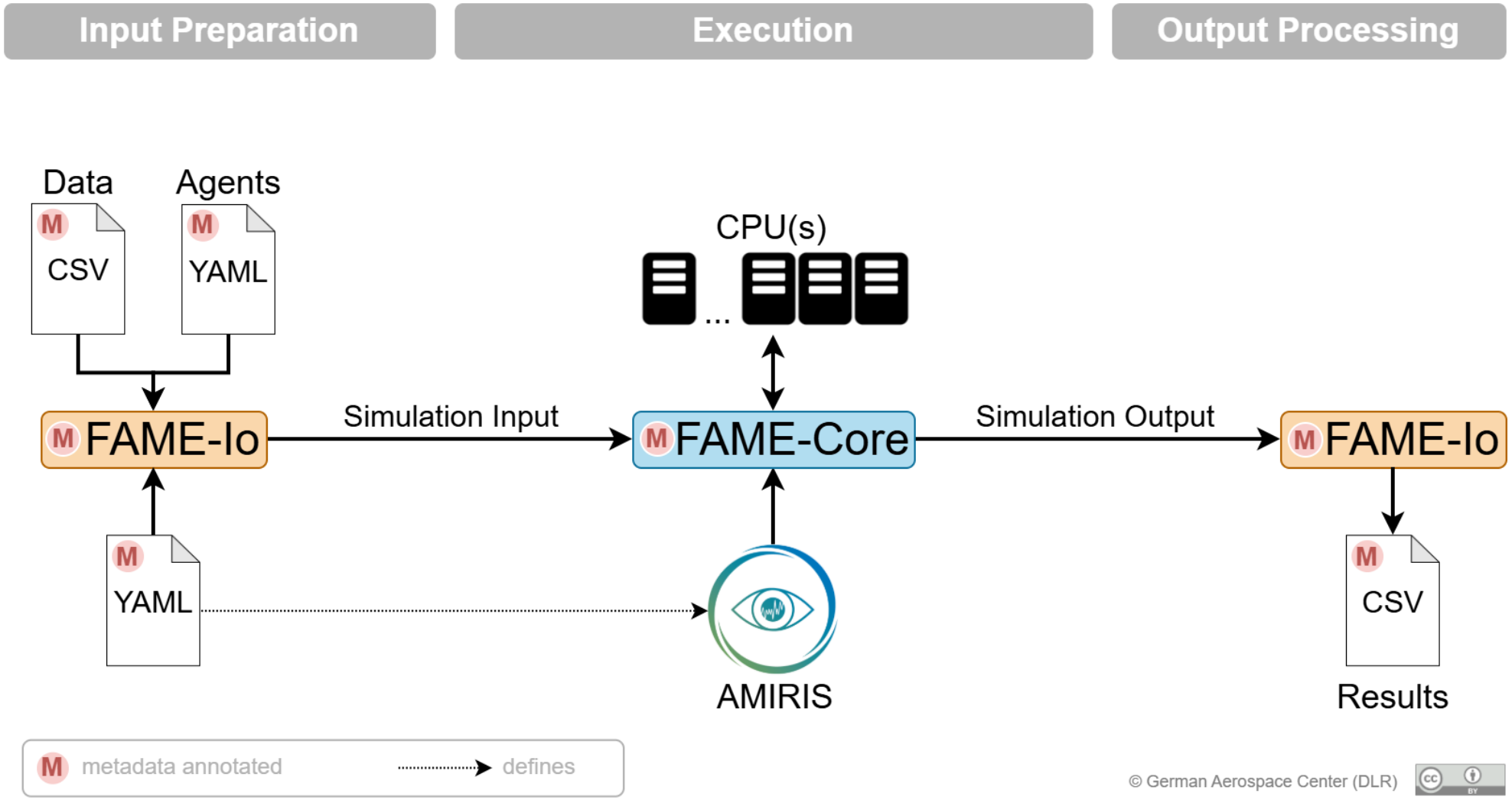
- Use same clearing as the DayAheadMarket
- Length of foresight window
- Importance of initial estimate
- Number of time steps until weight decays to 1/e
- Safeguard against accidentally small awards

Hands-on: AMIRIS Setup

Image source: DLR e.V.

AMIRIS: based on FAME

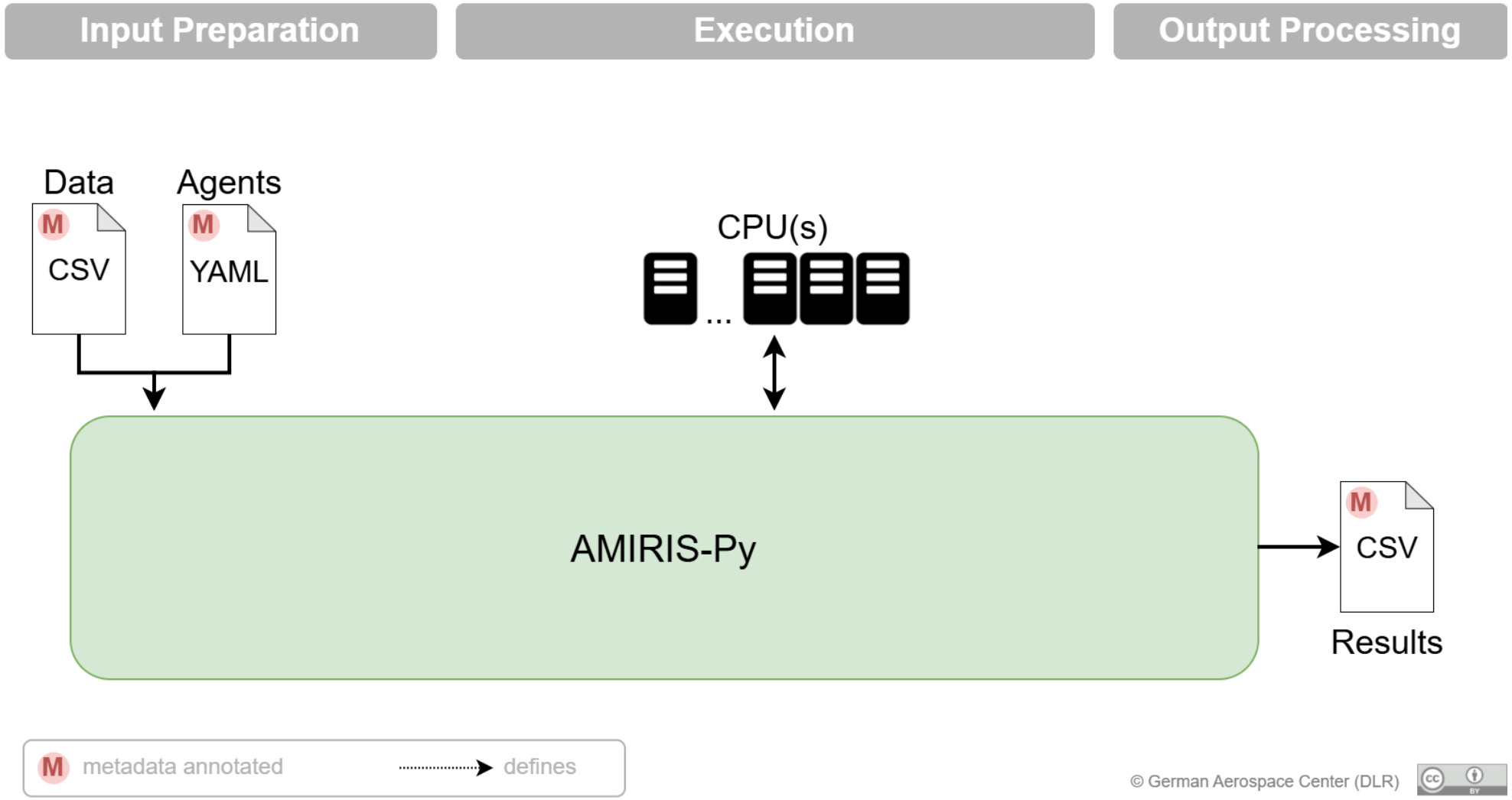
*The open **F**ramework for distributed **A**gent-based **M**odelling of **E**nergy systems*



FAME-Core: https://joss.theoj.org/papers/10.21105/joss.05087
FAME-Io: https://joss.theoj.org/papers/10.21105/joss.04958

AMIRIS: based on FAME

*The open **F**ramework for distributed **A**gent-based **M**odelling of **E**nergy systems*



FAME-Core: https://joss.theoj.org/papers/10.21105/joss.05087
FAME-Io: https://joss.theoj.org/papers/10.21105/joss.04958

Setup
Requirements

- Java JDK 11

```
(base) PS C:\> java --version
openjdk 11.0.29 2025-10-21
OpenJDK Runtime Environment Temurin-11.0.29+7 (build 11.0.29+7)
OpenJDK 64-Bit Server VM Temurin-11.0.29+7 (build 11.0.29+7, mixed mode)
```

  - Obtain from, e.g., https://adoptium.net/

- Python 3.10

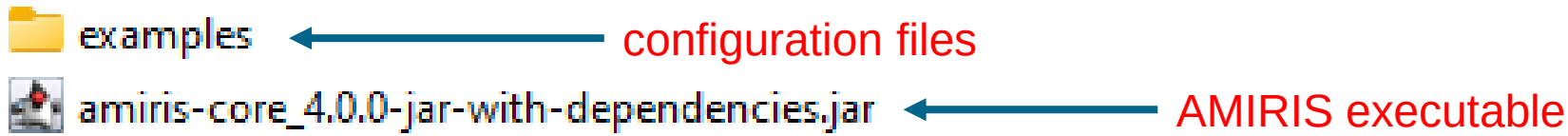
```
(base) PS C:\> python --version
Python 3.10.14
```

  - Obtain from, e.g., https://github.com/conda-forge/miniforge#mambaforge

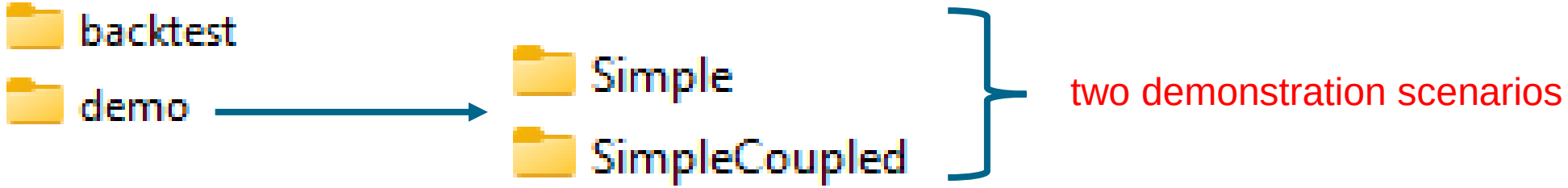
Setup
AMIRIS-Py

- Create environment
```
(base)> conda create -n AmirisEnv python=3.11
```
- Activate environment
```
(base)> conda activate AmirisEnv
```
- Install *amirispy*
```
(AmirisEnv)> pip install amirispy
```
- Create folder
```
(AmirisEnv)> mkdir amiris; cd amiris
```
- Download AMIRIS
```
(AmirisEnv)> amiris download
```

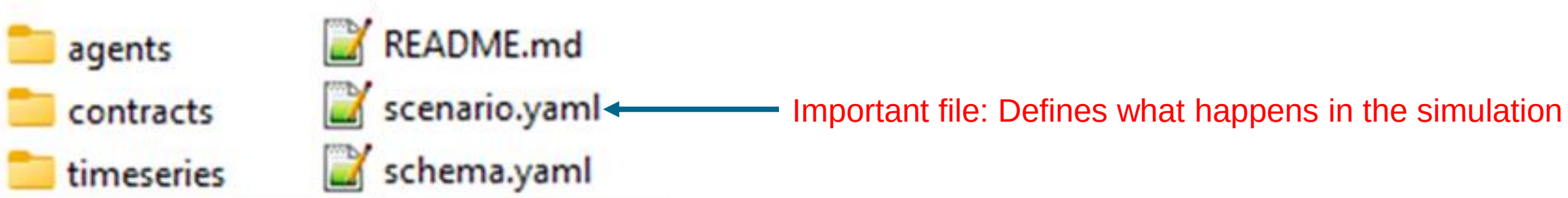
Setup
Files



examples/



examples/demo/Simple/



Setup
Run AMIRIS

```
(amirisEnv) PS C:\amiris> amiris run
usage: amiris run [-h] --scenario SCENARIO [--jar JAR] [--output OUTPUT]
                  [--output-options OUTPUT_OPTIONS] [--no-checks]
```

Required argument

- -s Scenario file

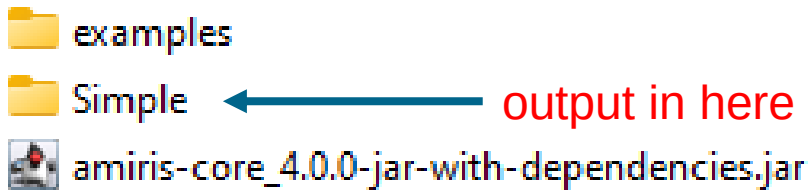
Use tab-completion for scenario

```
(amirisEnv) PS C:\amiris> amiris run –s .\examples\demo\Simple\scenario.yaml –o Simple
```

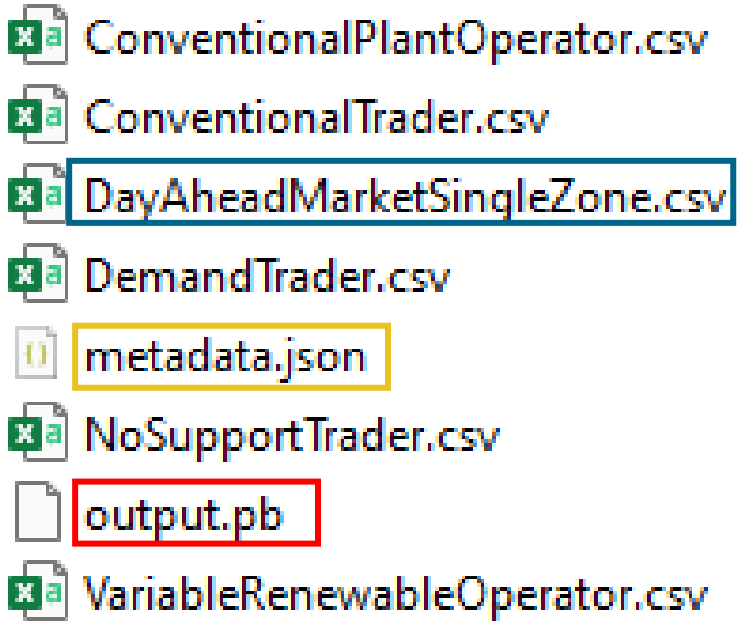
output folder

Console output

```
2026-03-27 13:47:21:: Simulation completed after executing 291 ticks in 0.16 seconds.
13:47:21 – PRINT – Successfully executed AMIRIS. See your results in ‘Simple'
```



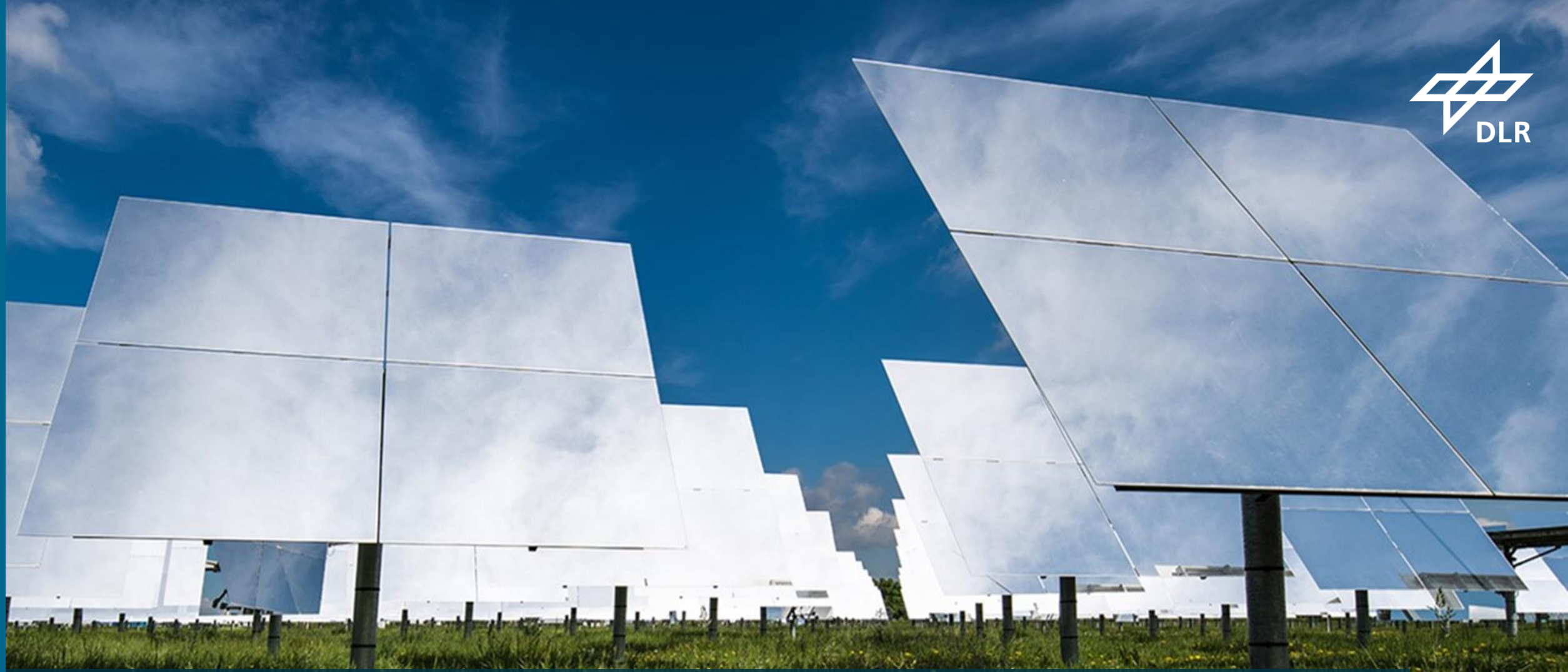
Setup
Results



AgentId	TimeStep	AwardedEnergyInMWH	ElectricityPriceInEURperMWH
1	01.01.2021 00:00	12431	267.4721054
1	01.01.2021 01:00	11416	262.9066734
1	01.01.2021 02:00	11163	260.8119727
1	01.01.2021 03:00	11036	257.4786831
1	01.01.2021 04:00	11192	256.4702082
1	01.01.2021 05:00	12177	256.2193284
1	01.01.2021 06:00	12685	256.2193284
1	01.01.2021 07:00	15222	259.7771467
1	01.01.2021 08:00	16491	260.2935264
1	01.01.2021 09:00	17125	257.9859146
1	01.01.2021 10:00	17378	255.7190453
1	01.01.2021 11:00	16997	255.4696391
1	01.01.2021 12:00	16237	257.2258181
1	01.01.2021 13:00	15476	256.4702082

Result binary: input, output, metadata

metada.json: fields described by Open Energy Ontology
https://openenergyplatform.org/ontology/



Parametrisation Basics

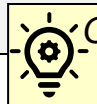
Parameterisation

Scenario: Main config file to bundle all simulation properties

Open: examples/backtest/Germany2018/scenario.yaml

scenario.yaml

Schema	*definition of valid agent and contract structures*
GeneralProperties	*simulation start/end time, random seed*
StringSets	*names used by multiple agents, e.g. fuels*
Agents	*which agents have what parameters*
Contracts	*how and when agents interact*
Metadata	*describing the scenario*



*Content of scenario.yaml distributed across separate files, for schema, agents, and contracts and joined with **!include**, see https://gitlab.com/fame-framework/fame-io#split-and-join-multiple-yaml-files*

Agents: !include ["agents/*.yaml", "Agents"]

Join contents in section “Agents” of each file ending with “.yaml” in folder “agents” and assign to field **Agents**.

Parameterisation
General Properties

- Define
  - start and end of simulation
  - which random seed to use

```
GeneralProperties:
  RunId: 1
  Simulation:
    StartTime: 2017-12-31_23:58:00
    StopTime: 2018-12-31_23:58:00
    RandomSeed: 1
```

ignore

ignore

*FAME's time definition **always** uses 365 days / 8760 hours per year, see also https://gitlab.com/fame-framework/wiki/-/wikis/architecture/decisions/TimeStamp*

YAML is indentation-based (2 spaces)

Parameterisation

Agents

Open: examples/backtest/Germany2018/agents/MarketsAndForecast.yaml

In YAML, dash is used to denote lists

- Define
  - agents
  - their type, ID, and attributes.
- Supported data types:
  - integer, floating point, enums, timeseries
- Supported structures
  - Any combination of blocks and lists
- Structure of attributes
  - depends on type of agent
  - is defined in schema

```yaml
Agents:
  - Type: DayAheadMarketSingleZone
    Id: 1
    Attributes:
      Clearing: &clearingParameters
        DistributionMethod: SAME_SHARES
      GateClosureInfoOffsetInSeconds: 31
  - Type: CarbonMarket
    Id: 3
    Attributes:
      OperationMode: FIXED
      Co2Prices: "./timeseries/co2_price.csv"
  - Type: FuelsMarket
    Id: 4
    Attributes:
      FuelPrices:
        - FuelType: LIGNITE
          Price: 5.00
          ConversionFactor: 1.0
        - FuelType: NATURAL_GAS
          Price: "./timeseries/natural_gas_cost.csv"
          ConversionFactor: 1.0
```

Time series attributes also support a single value.

*Every agent **must** have a unique ID within the simulation. This is how agents address each other.*

Single Storage

Image source: DLR e.V.

Single Storage
Germany 2018

```
(amirisEnv) PS C:\amiris> amiris run -s .\examples\backtest\Germany2018\scenario.yaml
                          -o 2018
```

Console output

```
2026-03-27 14:05:21:: Simulation completed after executing 184,027 ticks
in 11.76 seconds.
14:05:27 – PRINT – Successfully executed AMIRIS. See your results in ‘2018'
```

Open: 2018/ GenericFlexibilityTrader.csv

Agent Id	TimeStep	ReceivedMoney InEUR	StoredEnergy InMWH	VariableCosts InEUR	DispatchMultiplier	OfferedEnergy InMWH	AwardedEnergy InMWH
7	01.01.2018 00:00	0	40	0	1	854.4	-854.4
7	01.01.2018 01:00	0	40	0	1	0	0

Profit: 80.9 M€ **Dispatched energy**: 7.3 TWh **Average Price**: 46.22 €/MWh

Single Storage
Germany 2018

./examples/backtest/Germany2018/agents/Storage.yaml

```yaml
- Type: GenericFlexibilityTrader # Pumped Hydro
  Id: 7
  Attributes:
    Device:
      GrossChargingPowerInMW: 8988.8
      NetDischargingPowerInMW: 7120.
      ChargingEfficiency: 0.89
      DischargingEfficiency: 0.89
      EnergyContentUpperLimitInMWH: 40000.
      InitialEnergyContentInMWH: 1000.
    Assessment:
      Type: MIN_SYSTEM_COST
    StateDiscretisation:
      Type: STATE_OF_CHARGE
      PlanningHorizonInHours: 168
      EnergyResolutionInMWH: 160
    Bidding:
      Type: ENSURE_DISPATCH
      SchedulingHorizonInHours: 24
```

change assessment function to **MAX_PROFIT**

Single Storage
Profit Maximisation

```
(amirisEnv) PS C:\amiris> amiris run -s .\examples\backtest\Germany2018\scenario.yaml
                          -o MaxProfit
```

Console output

```
2026-03-27 14:05:21:: Simulation completed after executing 184,027 ticks
in 11.76 seconds.
14:05:27 – PRINT – Successfully executed AMIRIS. See your results in ‘MaxProfit'
```

Open: MaxProfit/GenericFlexibilityTrader.csv

Open: MaxProfit/DayAheadMarketSingleZone.csv

Profit: 114.0 M€ **Dispatched energy**: 4.8 TWh **Average Price**: 46.30 €/MWh

Competing Storage Units

Image source: DLR e.V.

Competing Storage Units
Germany 2019

`./examples/backtest/Germany2019/agents/Storage.yaml`

```
Agents:
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
- Type: GenericFlexibilityTrader
```

18 pumped-hydro storage units, differing by

- Efficiency
- Converter power
- Reservoir size

Details

Schimeczek et al. (2026). 10.1016/j.est.2025.120054

Competing Storage Units
Germany 2019

```
(amirisEnv) PS C:\amiris> amiris run -s .\examples\backtest\Germany2019\scenario.yaml
                          -o 2019
```

Console output

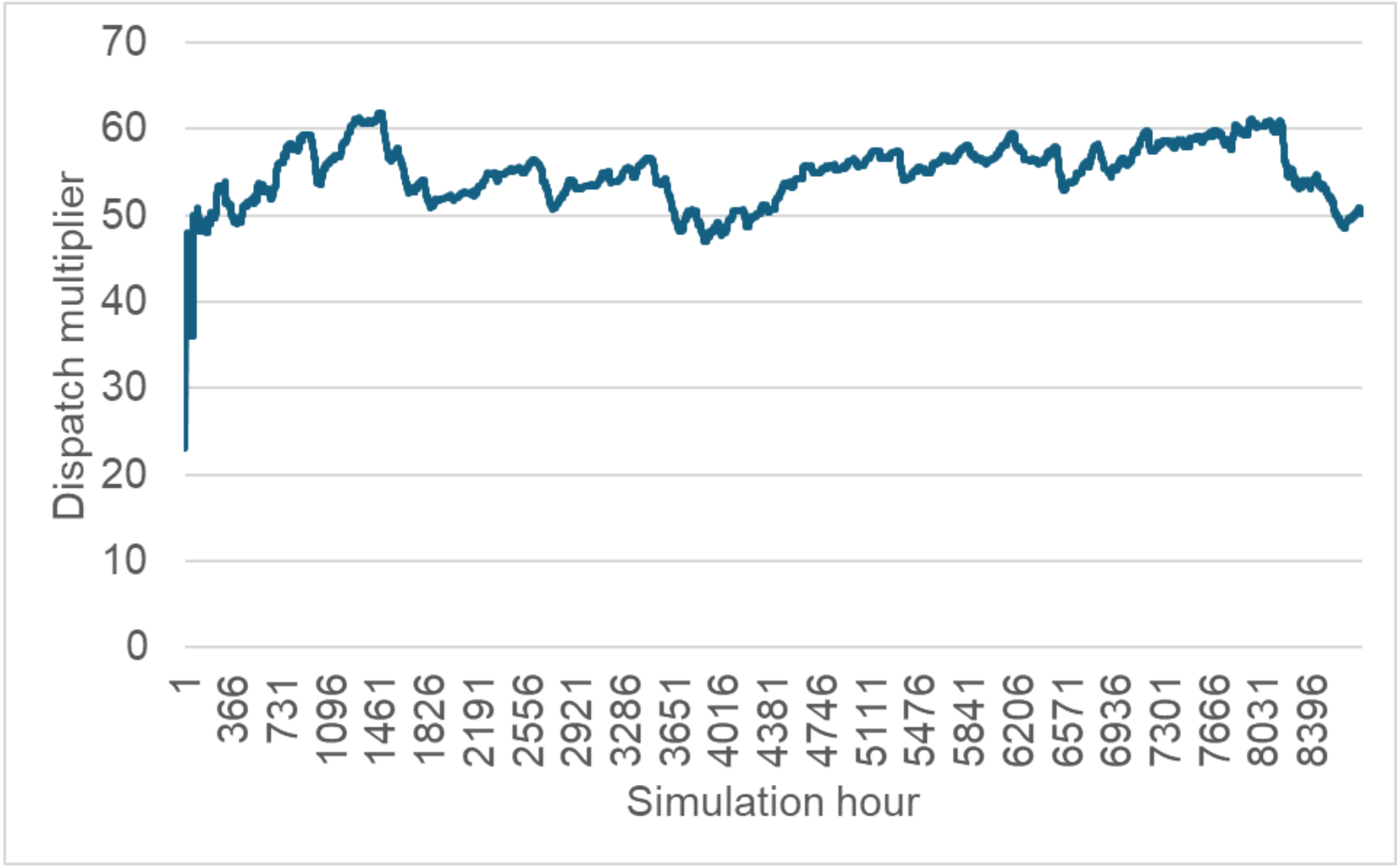
```
2026-03-27 17:37:07:: Simulation completed after executing 184,027 ticks
in 33.05 seconds.
17:37:14 – PRINT – Successfully executed AMIRIS. See your results in ‘2019'
```

Open: 2019/GenericFlexibilityTrader.csv

Competing Storage Units
Dispatch Multipliers

Agent 700
Small converter
→ High multiplier

Small reservoir
→ Even higher multiplier

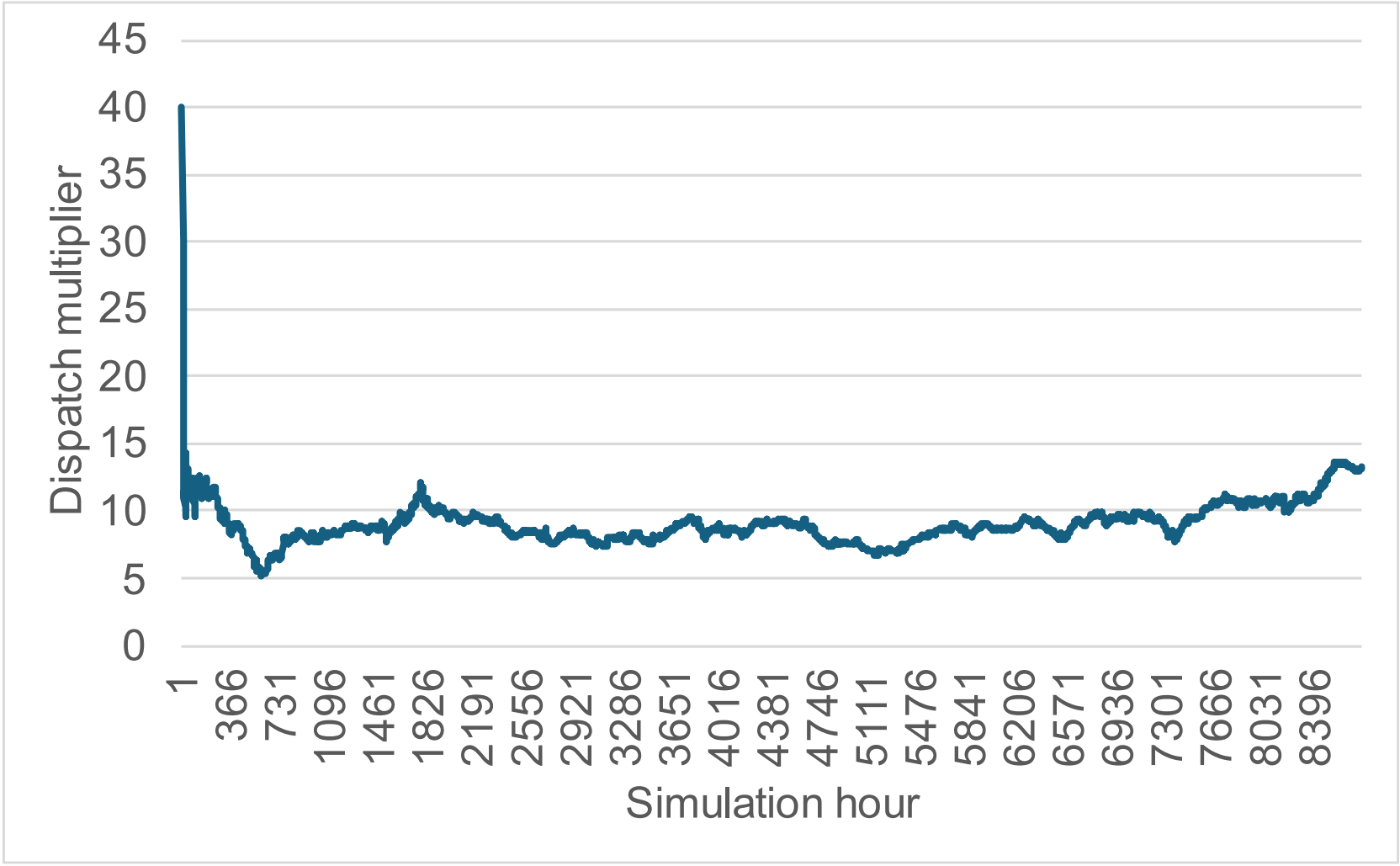


Competing Storage Units
Dispatch Multipliers

Agent 700
Small converter
→ High multiplier

Small reservoir
→ Even higher multiplier

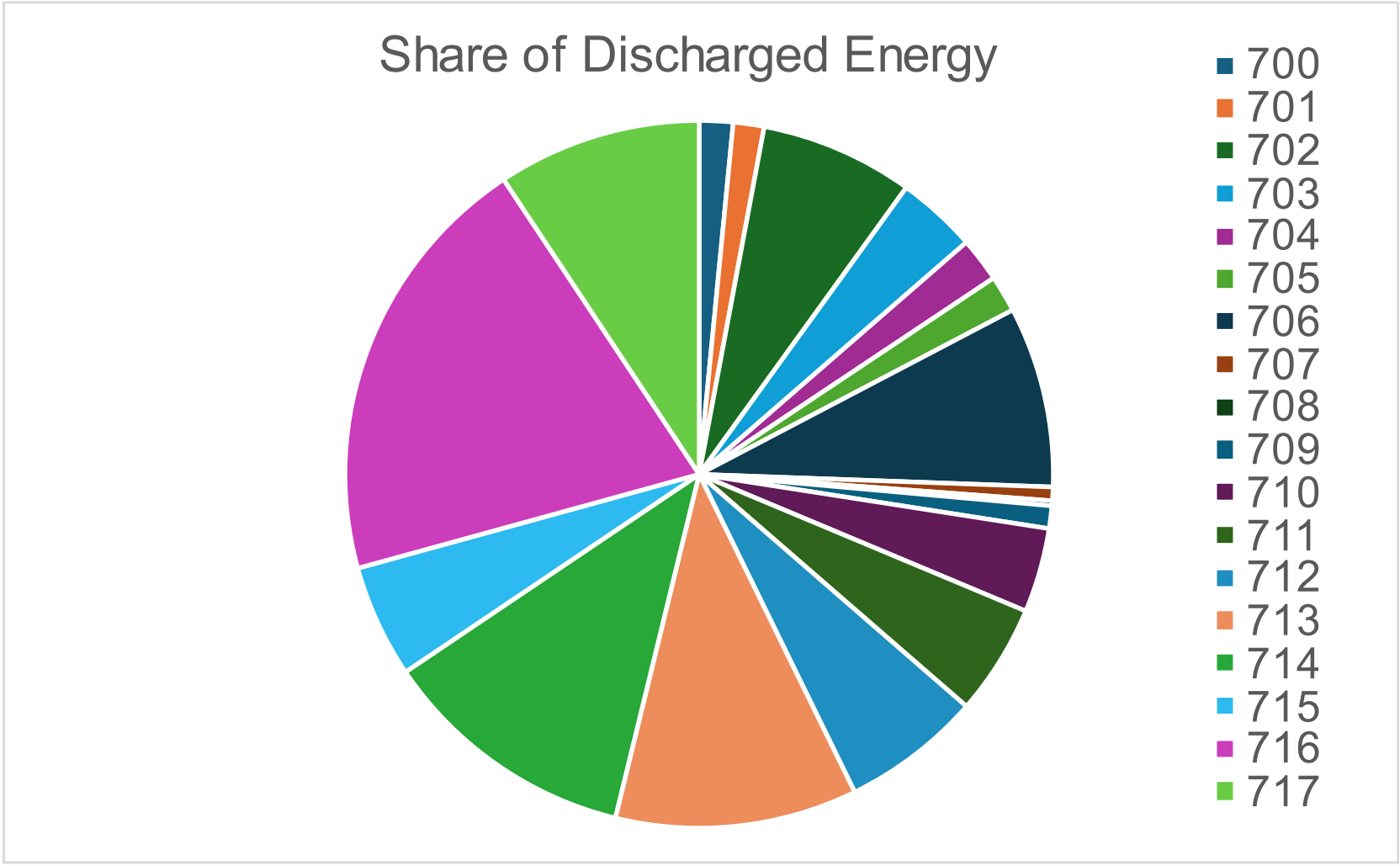
Agent 717
Even smaller converter
but huge reservoir
→ Significantly lower multiplier



Competing Storage Units
Storage Performance

Energy Share

- Depends on converter power, reservoir size and round-trip efficiency



Competing Storage Units

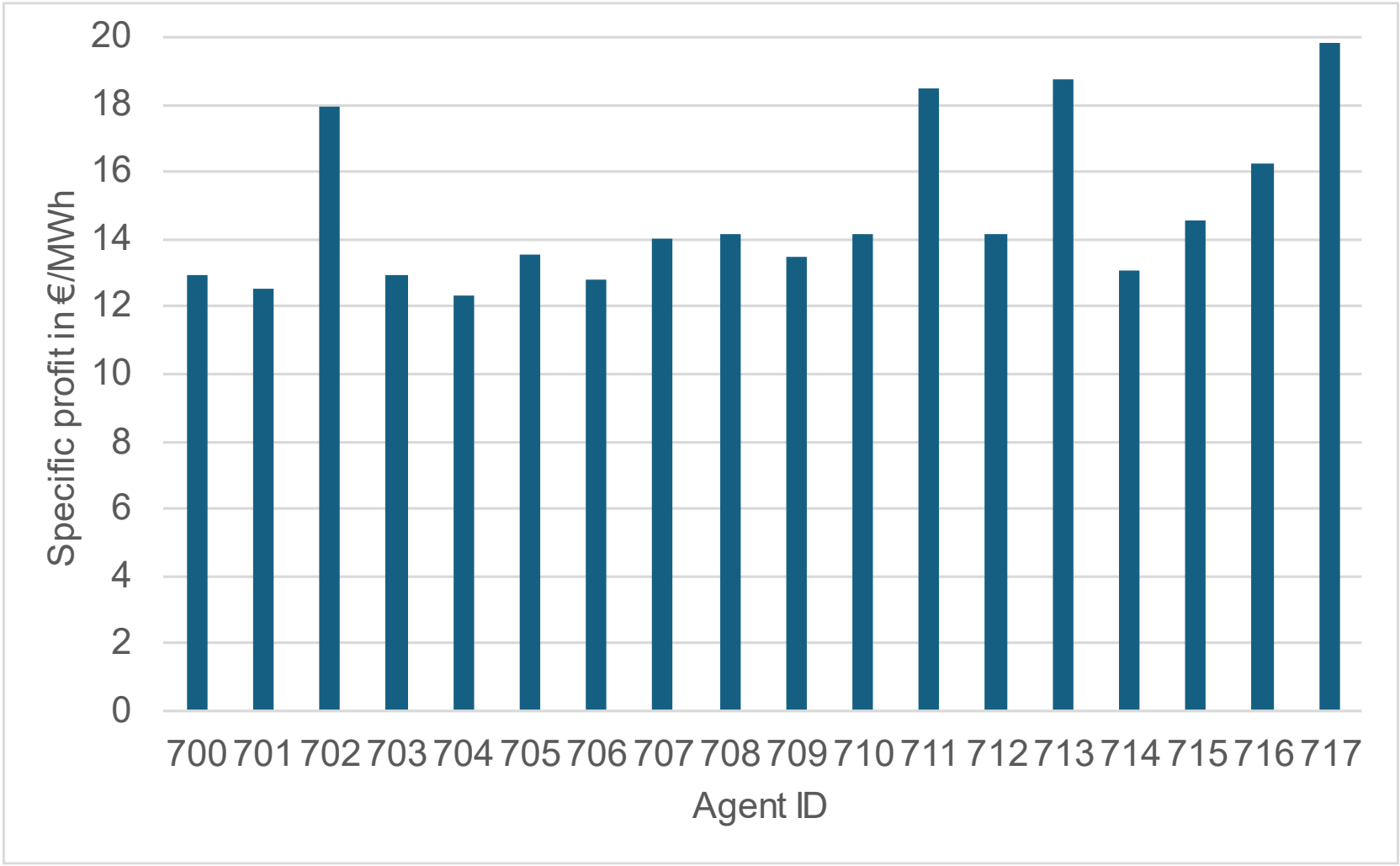
Storage Performance

Energy Share

- Depends on converter power, reservoir size and round-trip efficiency

Profit per MWh

- Higher margins for units with high round-trip efficiency



Add Capacity

Image source: DLR e.V.

Add Capacity
Edit Scenario

./examples/backtest/Germany2019/agents/Storage.yaml

```yaml
- Type: GenericFlexibilityTrader
  Id: 718
  Attributes:
    Assessment:
      Type: MIN_SYSTEM_COST
    Bidding:
      SchedulingHorizonInHours: 24
      Type: ENSURE_DISPATCH
    Device:
      ChargingEfficiency: 0.95
      DischargingEfficiency: 0.95
      EnergyContentUpperLimitInMWH: 10000.0
      GrossChargingPowerInMW: 10000.0
      InitialEnergyContentInMWH: 5000.0
      NetDischargingPowerInMW: 10000.0
    StateDiscretisation:
      EnergyResolutionInMWH: 200
      PlanningHorizonInHours: 168
      Type: STATE OF CHARGE
```

Add large battery storage

- Copy / paste from 717
- Assign Id: 718
- Increase efficiency
- 10 GW / 10 GWh
- Reduce planning horizon
- Increase EnergyResolution to 200 MWh

Add Capacity
Add contracts

./examples/backtest/Germany2019/contracts/storage.yaml

```
AgentGroups:
  - &exchange 1
  - &forecaster 6
  - &storage [700, 701, 702, 703, 704, 705, 706, 707, 708, 709, 710,
```

Add new agent 718 to storage group

Add Capacity
Re-run simulation

```
(amirisEnv) PS C:\amiris> amiris run -s .\examples\backtest\Germany2019\scenario.yaml
                          -o MoreStorage
```

Console output

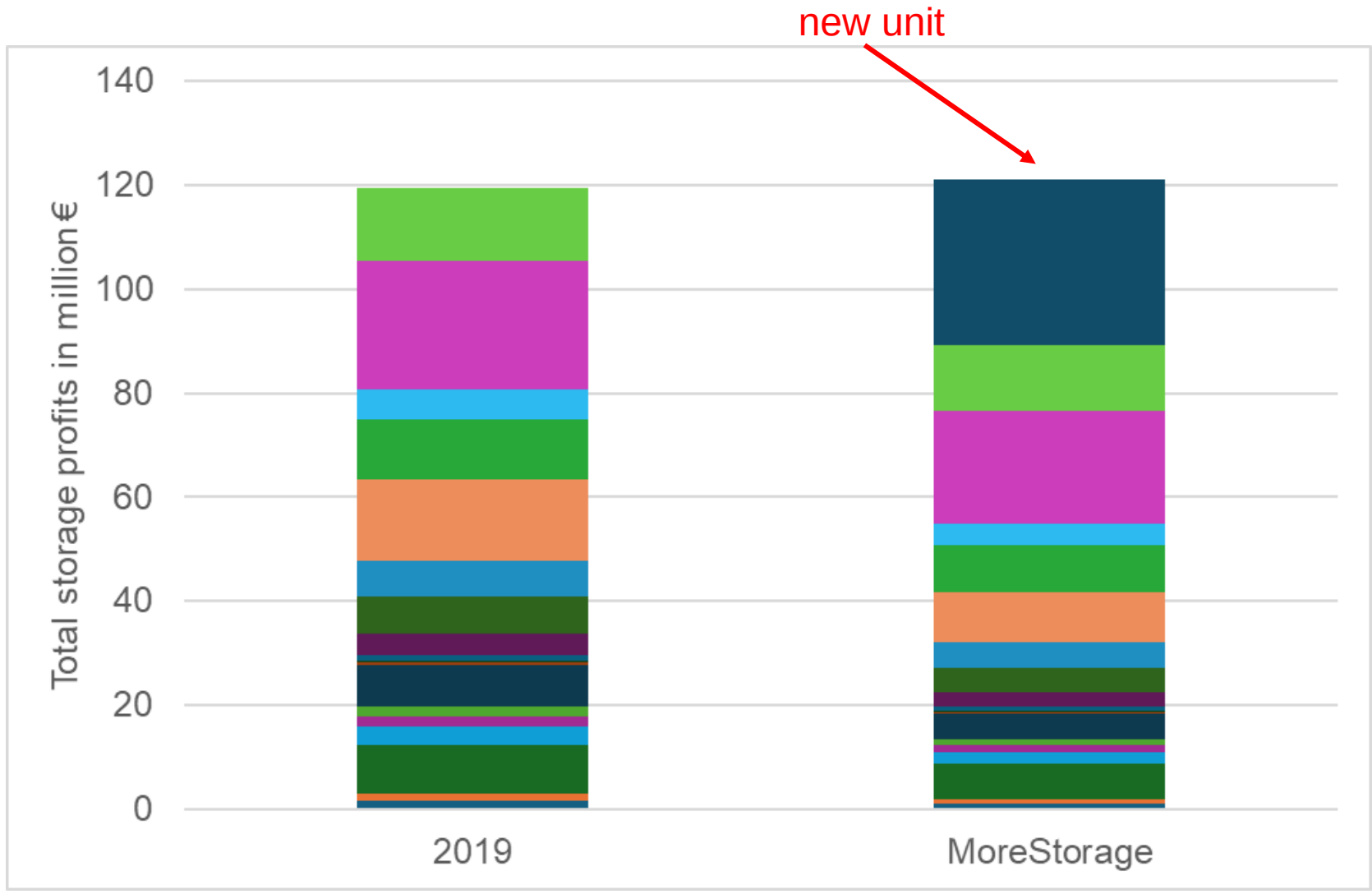
```
2026-03-27 18:12:01:: Simulation completed after executing 184,027 ticks
in 33.05 seconds.
18:12:10 – PRINT – Successfully executed AMIRIS. See your results in ‘MoreStorage'
```

Open: MoreStorage/GenericFlexibilityTrader.csv

Add Capacity
Profitability Assessment

Profit
- New unit takes large share of total profits
- Total profit almost the same

→ cannibalisation



Outlook: AMIRIS 4.1

Image source: DLR e.V.

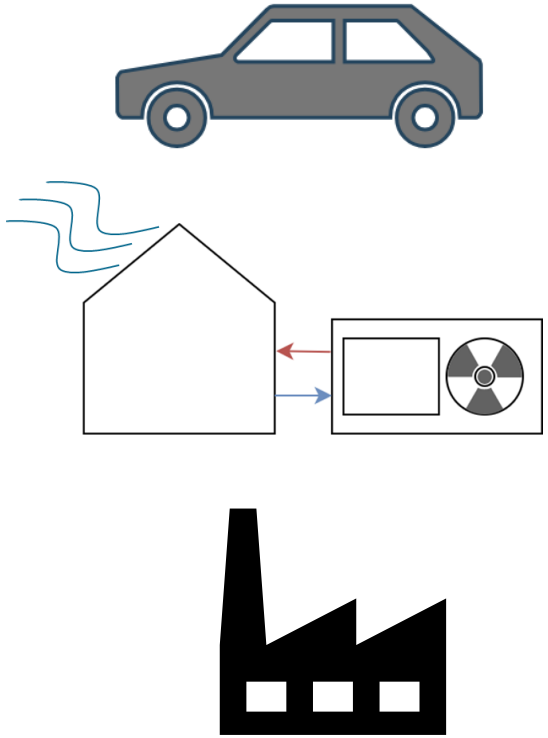
Outlook: AMIRIS 4.1

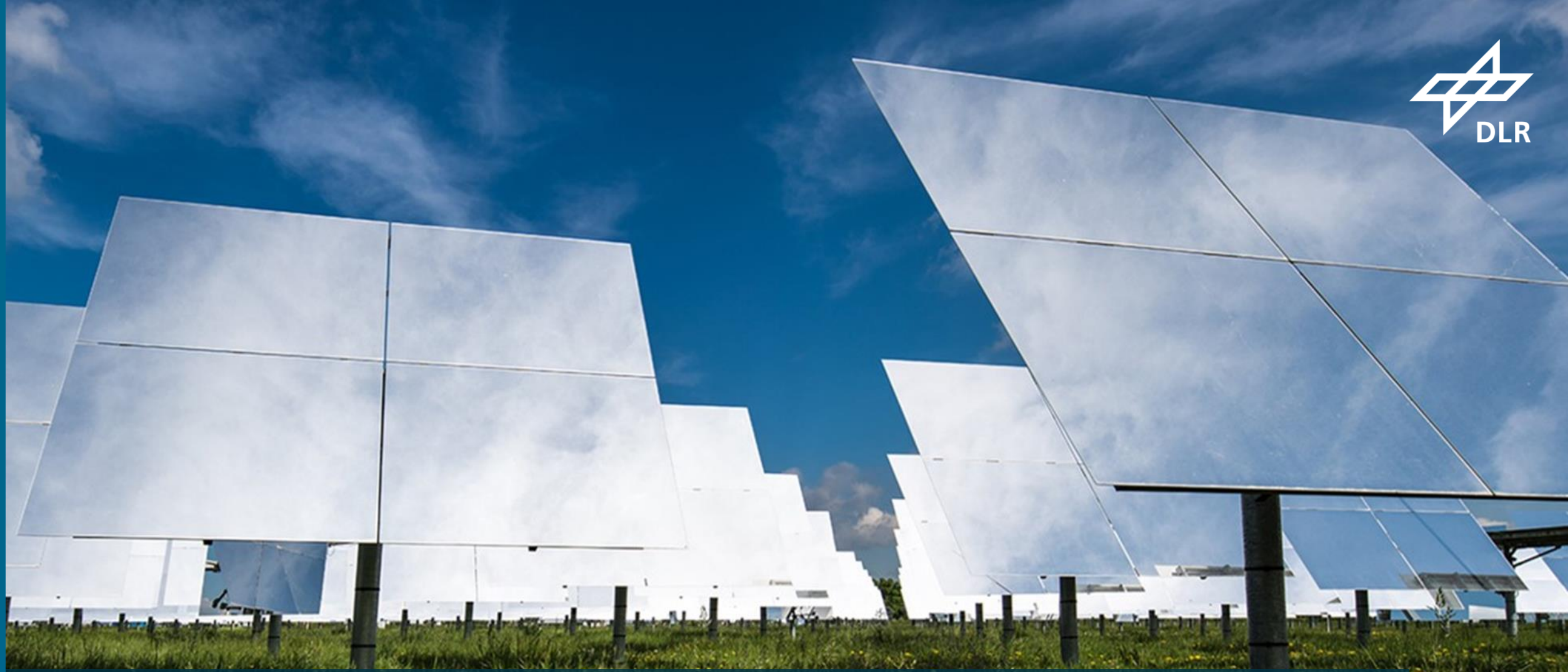
Sector-coupling Flexibility Options

Electric vehicles
- Grid connection rate
- Consumption from driving
- Unidirectional & bidirectional charging

Heat pumps
- Temperature loss profile
- Performance factor profile
- Accepted temperature levels

Load shifting
- Maximum shift time
- Time-dependent capacities and reservoirs
- Shift-time prolonging

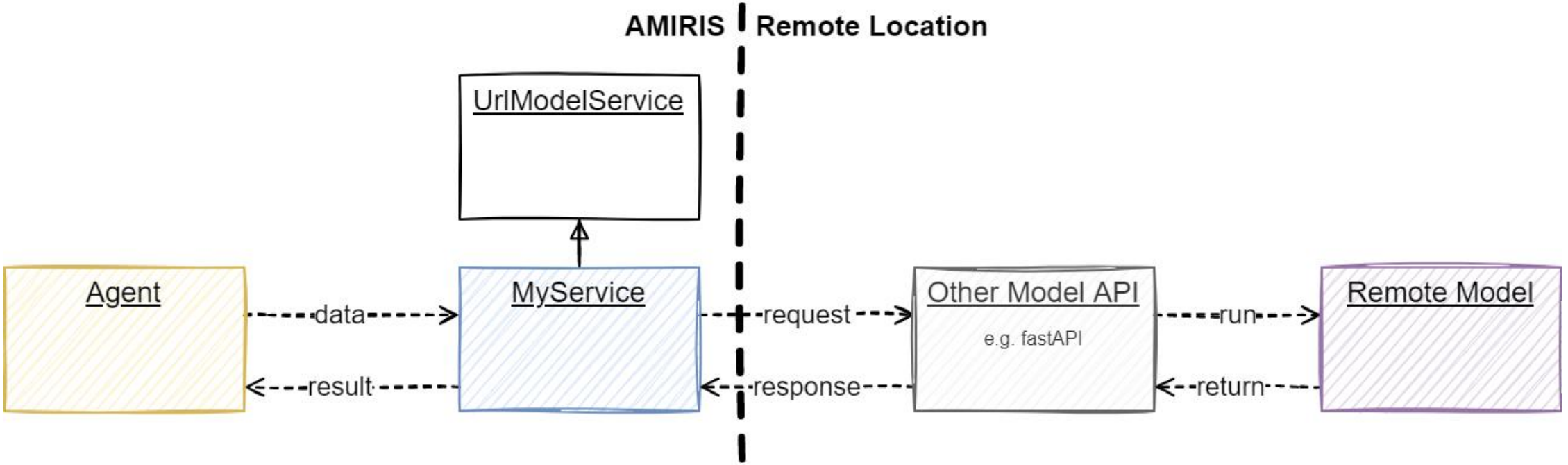




Final Remarks

AMIRIS: Model Coupling

Support tools



Benefits

- Easy setup
- Couple with virtually any model
- Allows *interactive* coupling

Examples

- Load shifting optimisation
- Market clearing model
- Heat pump operation
- Neural Network Decision Modules

AMIRIS

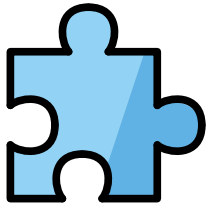
Following FAIR4RS Principles

Findable

- Website
- DOI
- Wikipedia
- COMSES
- HECI
- OEP
- openmod

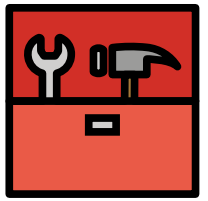
Accessible

- GitLab
- PyPI
- Zenodo



Interoperable

- API
- Workflow tools
- CSV
- YAML



(Re-)usable

- Apache 2.0
- REUSE
- Wiki
- Javadoc
- Win/Mac/Linux
- Scalable (H)PC

Key Indicators

Users

- 35 confirmed external user
- 15 external contributions

PhD (candidates)

- 8 internal
- 7 external

Visibility

- 34k views on Wikipedia
- 28k views on openmod

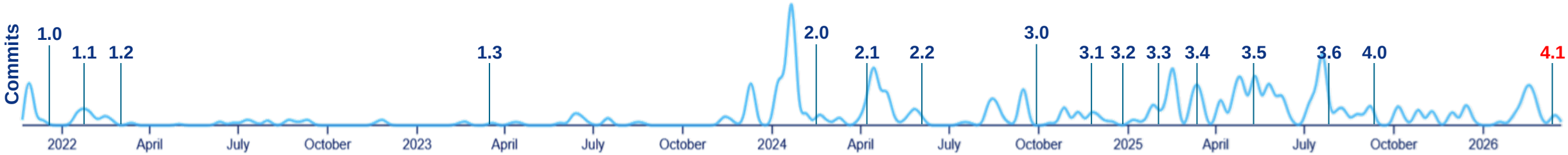
Software

- 51 releases
- 49k downloads

Barker et al. (2022). 10.1038/s41597-022-01710-x

AMIRIS

Activity & Community Support



1237 commits | Last commit ≈ 5 days ago | 19 stars | 11 forks

Post any question about AMIRIS at the openMod Forum.

Send us an email.

Discuss AMIRIS issues at our Open Forum, every Friday 10 o'clock CET.

Raise issues and feel free to contribute.

Imprint

Topic	**How to Model Competing Flexibility Options (Fast)**
Date	2026-03-31
Authors	Christoph Schimeczek, Felix Nitsch, Johannes Kochems, Kristina Nienhaus
Contact	amiris@dlr.de
Institute	German Aerospace Center Institute of Networked Energy Systems
Licenses	DLR (CC BY 4.0), except DLR Logo, background image, AMIRIS Logo © DLR