



Gossip protocol approach for a decentralized energy market with OPC UA client-server communication

Mr. Schindler Josef, Framatome GmbH / Friedrich-Alexander-University Erlangen-Nuremberg

Ms. Tellabi Asmaa, Framatome GmbH / University of Siegen

Dr. Waedt Karl, Framatome GmbH

Virtual meeting: 5th GI/ACM I4.0 Standardization Workshop on Industrial Automation and Control Systems (IACS) workshop, 28 September 2020

SUMMARY

1. Background story – DECENT
2. OPC UA
3. Gossip – Basics
4. Implementation
5. Simulation & Results
6. Conclusion

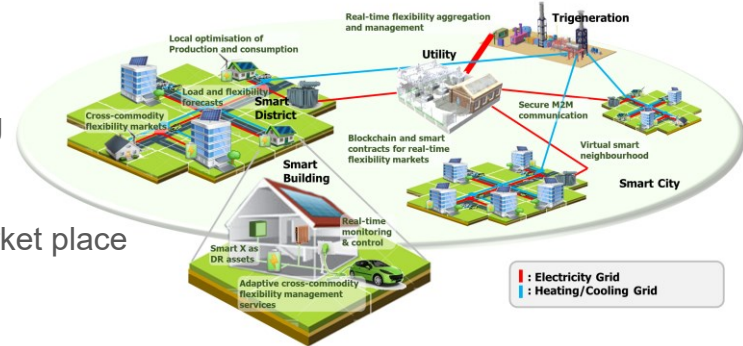
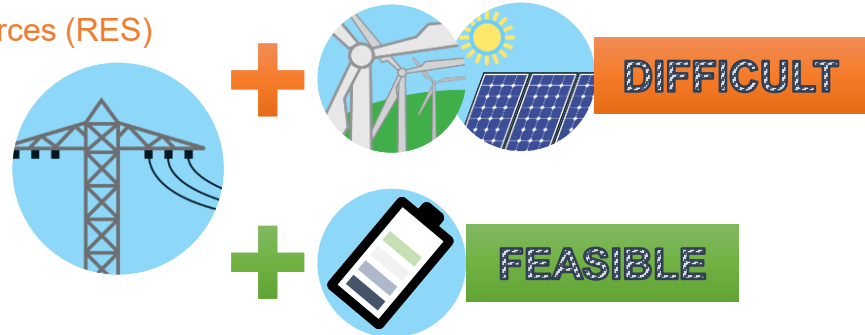
Background story – DECENT

- Volatile, non-controllable Renewable Energy Sources (RES)
 - bad influence on power grid
 - Flexibilities needed (additionally)
 - Idea: linkage of commodities

- Example:
 - Storing of electricity is very difficult
 - Storing of heat is easy (e.g. inside water)
 - Indeed losses as heat storing
 - Perfect for short-term scenarios!

→ DECENT project

- Market driven, holistic approach for cross-commodity sharing
- Project implementation: merit order market, blockchain for trust-management and memorizing smart contracts
- Research of this paper: gossip network as an alternative market place



OPC UA – Basics

- Industrial Machine to Machine (M2M) communication protocol for interoperability
 - for wireless and wired systems
 - M2M → integral part of the Internet of Things (IoT)
- Cross-platform Service Oriented Architecture (SOA) for Process Control
 - Enhanced Security, based on new standards
 - Based on different logical levels
- Provision of an Information Model
 - Full Mesh Network based on Nodes
 - Nodes → Processing of Data and Metadata

OPC UA – Standardization

► Core Specification Parts

1. Overview & Concepts
2. Security Model
3. **Address Space Model**
4. Services
5. **Information Model**
6. Service Mappings
7. Profiles

► Access Type Specification Parts

1. Data Access
2. Alarms & Conditions
3. Programs
4. Historical Access

► Utility Specification Parts

1. Discovery
2. Aggregates

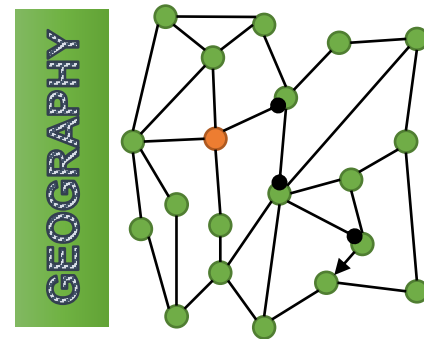
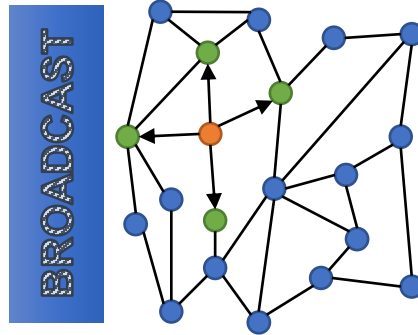
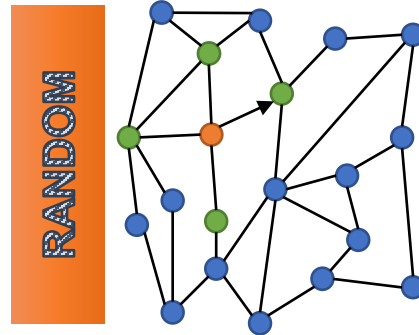
OPC UA – Library: FreeOPCUA

- Open-source library
- Creates OPC UA servers and clients based on Python and C++ Languages
- Supports most of the operations such as read, write, explore...
- Compatible with Linux based and Windows based systems

➔ **Some dependencies are required for the library to be installed**

Gossip – Basics

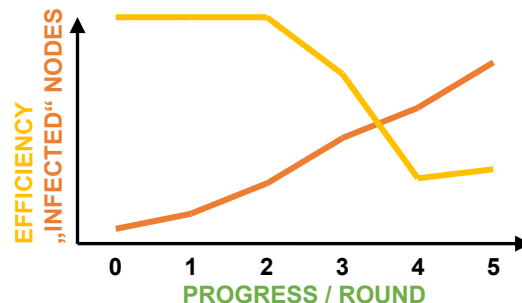
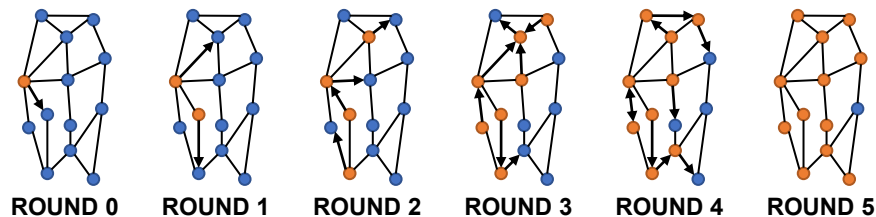
- Communication network
- Idea: spreading information to neighbours
- Types
 - **Random** → a single randomly chosen neighbour is addressed
 - **Geography** → nodes with a larger hop-limit than 1
 - **Broadcast** → all surrounding nodes
- Notes on examples (right side)
 - One round
 - One initial node shares information (orange)
 - Green nodes are accessible
 - Arrows highlight the targeted one(s)



Gossip – Basics – Example – Network type: random

Measures:

- **Progress** → One round = each infected node addresses one other node
- **Number of “infected” nodes** → measured after each round
- **Efficiency** → Newly “infected” nodes divided by total attempts



Efficiency:

- High at low degree of information spread
- Rather low with increasing knowledge



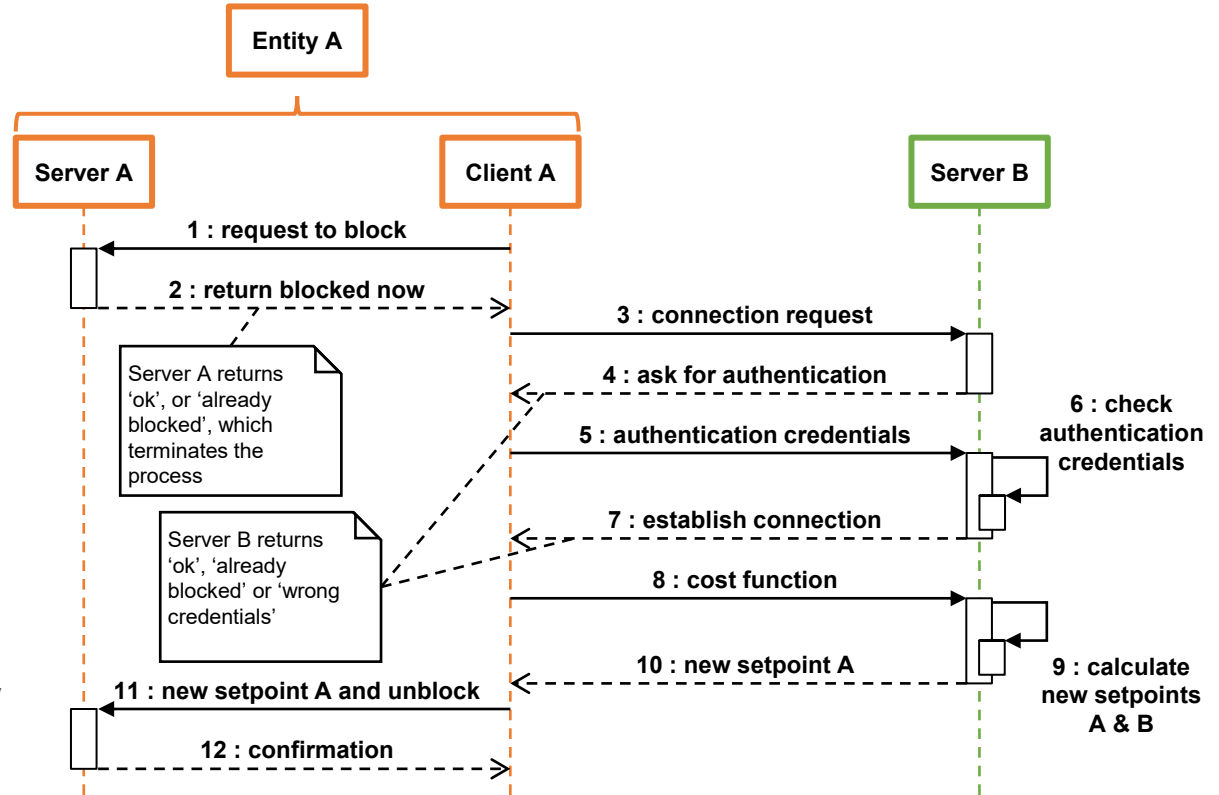
Good for a quick estimation / consensus



Not the best for the exact optimum

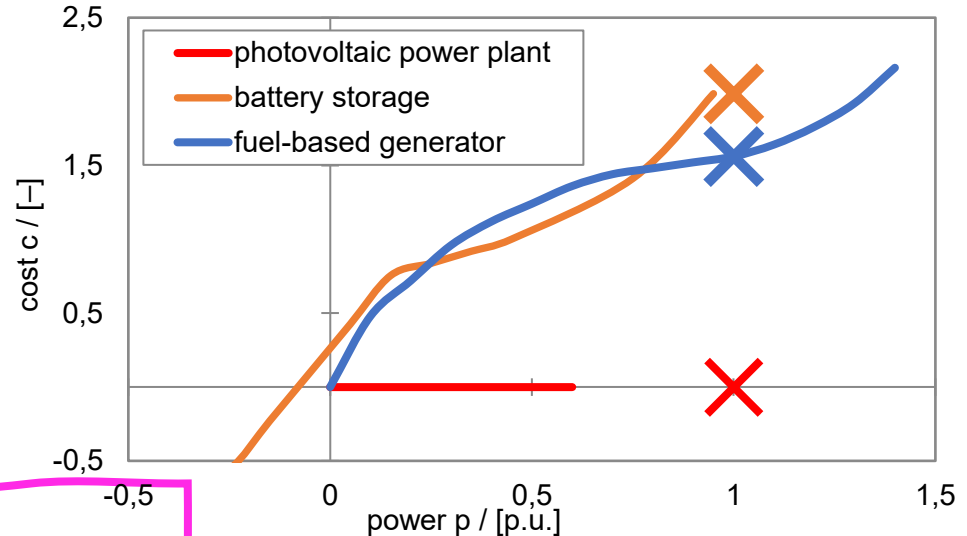
Implementation – Local consensus between two entities

- Example with Entity A & B
- No longer fixed rounds
(Entities wait for random time)
- Local consensus
 - Not just information sharing
 - Bidirectional communication
- **Entity A:** Internal check
- **Both:** Connection request
- **Both:** Authentication procedure
- **Both:** Consensus finding
- **Entity A:** Setting setpoint internally



Implementation – Cost function examples

- Local consensus:
 - Calculated in one gossip between 2 entities
 - Optimum of combination of 2 cost functions
- Examples on right side:
 - [p.u.]: per unit; relative to rated power
 - PV: no change in cost
 - Battery storage: negative values allowed
 - Fuel-based generator
 - Bad efficiency at low setpoints
→ high gradient
 - High wearing on components at high setpoints
→ high gradients



- Global consensus:**
 - Minimum overall cost with steady overall power output
 - Result from many local consensus findings

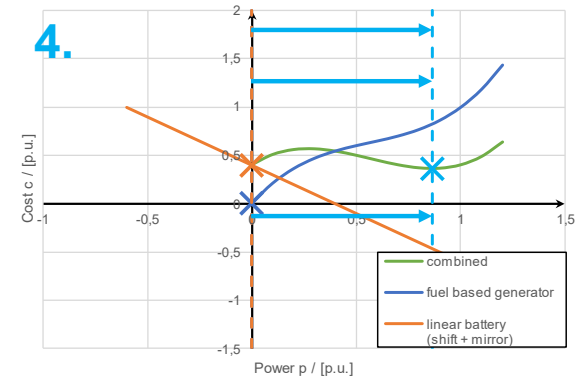
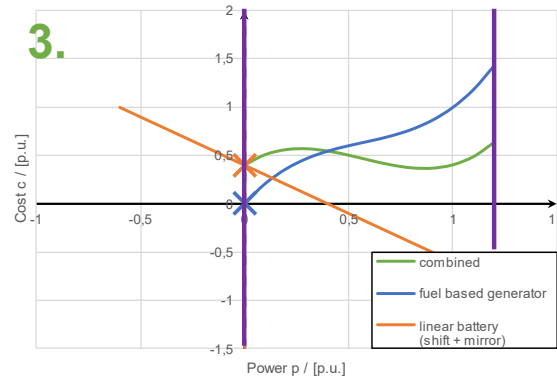
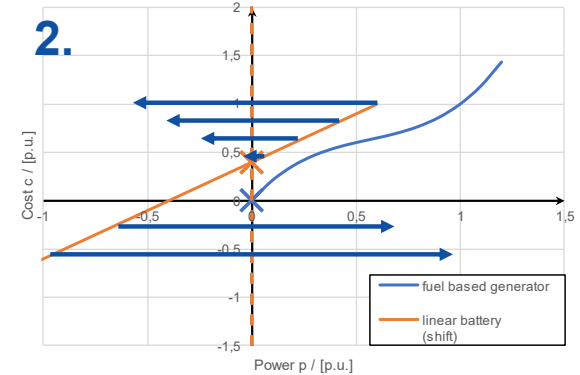
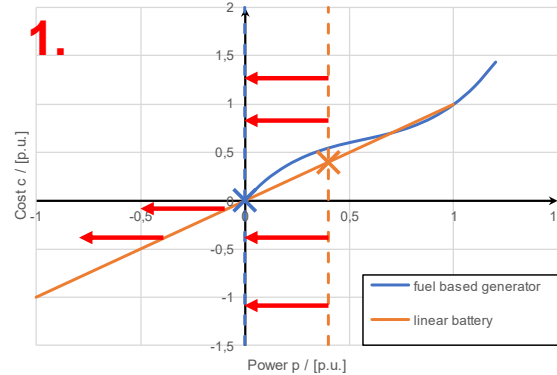
= GOAL

Implementation – Combination of 2 cost functions

- Polynomials + Setpoint + Boundaries

Process:

- Shift of battery**
(so both current setpoints are at 0)
- Mirror of battery**
 - Around y-axis
 - One entity increases setpoint \Leftrightarrow other decreases
- Combine both functions**
 - Just add all coefficients
 - Keep smaller boundaries**
- Calculate setpoint change**
 - Minimum of the combined cost function**
 - Cost, in example, slightly lower
 - Note: Negative change for **battery**



Implementation – Basic vs. OPC UA

Basic

- Python library for communication: `socket`
- For timing: `settimeout`
- In the meantime listening
- Buffer will be parsed at server-side to list of
 - Current setpoint
 - Boundaries
 - Coefficients for polynomial
- No authentication implemented and foreseen

➔ **Some errors occurred, due to communication inaccuracies**
(Overall power differed?!?)

OPC UA

- Python library: `FreeOpcUa`
- Scheduling with: `sched`
- In the meantime listening
- OPC UA function call
 - High level
 - Setpoint, boundaries and coefficients for polynomial as function parameters
- No authentication implemented, but foreseen

➔ **Reliable solution**

Simulation & Results

- Virtual machines → communication network
- Initialization of each individual node with fixed values
- Variation of
 - Waiting time-interval: D1, **D2** (exemplarily highlighted), D3, D4
 - Number of entities: 8, 16, 24
- Simulation time: 90 s
- Each setup / run: 3 times
- 2 Implementations → 72 simulations

= Delay scenario D2

Delay-scenario	D1	D2	D3	D4
d_{min}	0.01 s	0.5 s	1 s	2 s
d_{max}	0.3 s	1 s	1.5 s	3 s

Entity with IPv4 Addresses: 192.168.56.1<ID>																									
<ID>		01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
IP-Adress & Identifier	p_{init}	0.4	0.5	0.6	0.6	0.4	0.6	0.7	0.8	0.9	0.4	0.0	0.3	2.6	1.3	-1.0	0.8	4.3	0.3	0.2	3.3	4.3	-1.0	1.5	-1.0
	c_{init}	2.58	1.88	4.12	4.12	3.38	1.82	4.04	4.28	2.03	0.31	3.0	0.2	4.52	-2.45	1.9	-2.21	-1.35	0.69	3.51	1.82	3.3	1.8	3.04	-1.1
Initial values	p_{min}	-1.1	0.0	-2.0	-2.0	-1.5	0.6	-2.0	-2.8	0.1	-2.2	-1.0	-1.4	-1.4	-2.4	-2.2	-1.3	0.0	-2.2	-1.4	0.0	1.0	-2.0	-1.4	-1.4
	p_{max}	1.9	3.0	3.0	3.0	1.5	6.0	1.7	1.8	2.2	1.3	1.0	1.2	8.4	1.4	1.1	0.0	5.0	1.1	1.4	5.0	7.0	3.1	1.6	0.3
Boundary setpoints	a_0	1.0	2.0	4.0	4.0	1.0	2.0	4.0	4.0	2.0	0.5	3.0	0.2	4.0	-5.3	0.9	0.2	-3.1	0.6	3.2	0.3	-1.0	0.2	1.1	0.6
	a_1	3.0	-0.5	-0.3	-0.3	5.0	-0.6	-0.7	-0.8	-1.2	-0.4	2.5	0.2	0.2	1.6	-0.8	-5.3	2.3	0.1	1.6	-0.2	1.0	-1.2	0.6	0.3
Polynomial coefficients	a_2	2.0	0.5	0.6	0.6	2.0	0.5	0.6	0.8	1.1	-0.3	-5.2	-0.8		0.38	1.0	1.1	-1.3	0.5	-0.3	0.2	0.0	0.6	0.8	-1.1
	a_3	1.0		0.4	0.4	1.0		0.7	0.8	0.3	0.2	3.1	0.4		0.06	0.8	2.2	0.2	0.7	0.2			0.2	-1.2	0.5
	a_4										0.1		0.2						0.1					0.2	0.2
	a_5												-0.1											0.3	
	a_6												0.1												

Gossip – Basics – Conclusion

RECAP

Efficiency:

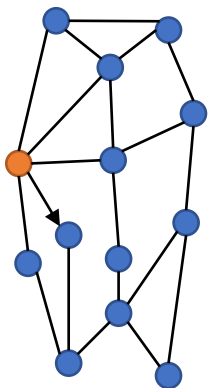
- High at low degree of information spread
- Rather low with increasing knowledge



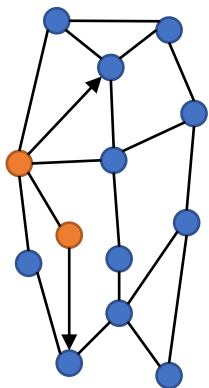
Good for a quick estimation / consensus



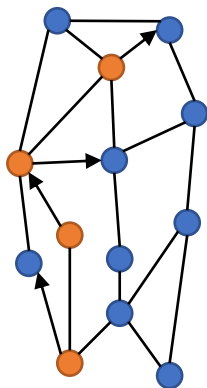
Not the best for the exact optimum



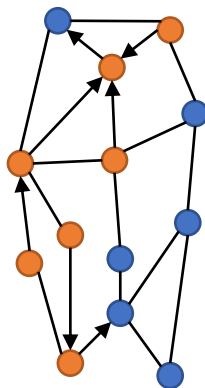
ROUND 0



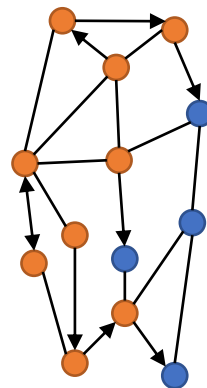
ROUND 1



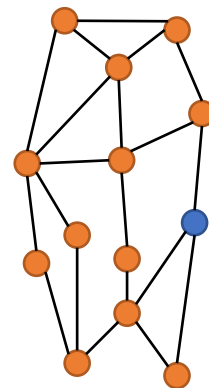
ROUND 2



ROUND 3



ROUND 4



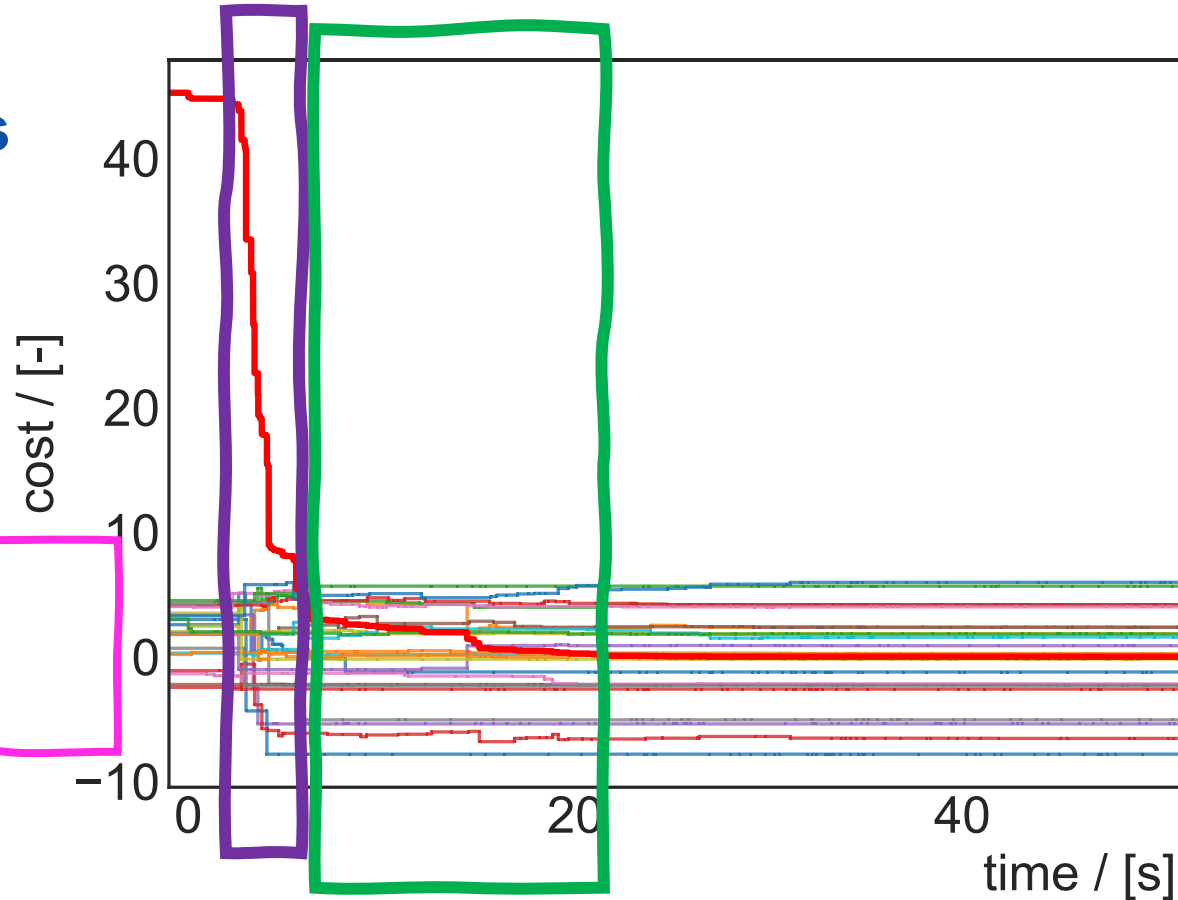
ROUND 5

Simulation & Results

- Exemplarily (OPC UA):
 - $N = 24$
 - $d_{min} = 0.01$ s
 - $d_{max} = 0.3$ s
- Overall (& individual) cost
- Weigh shorter than 90 s

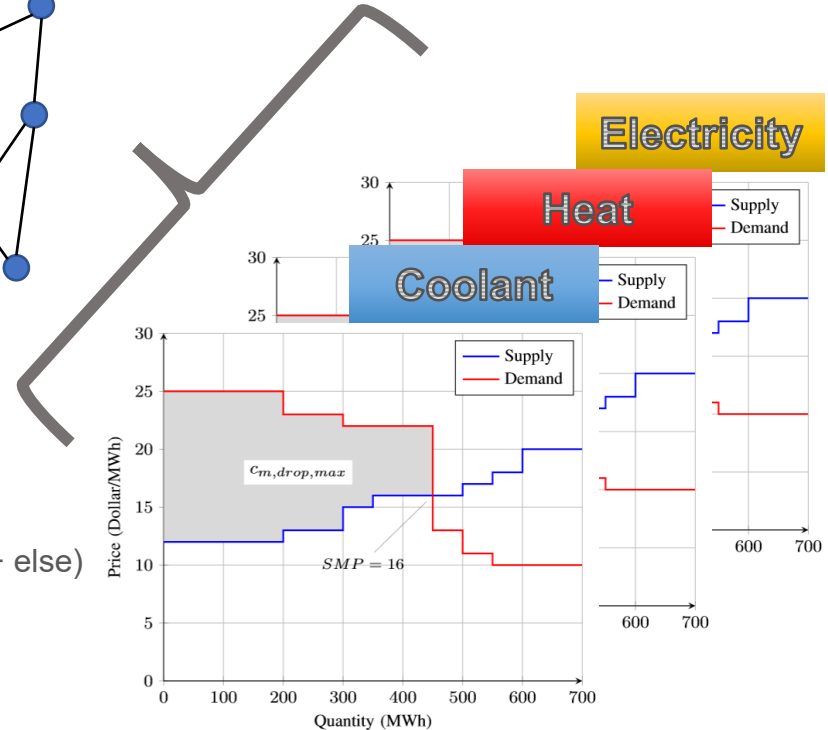
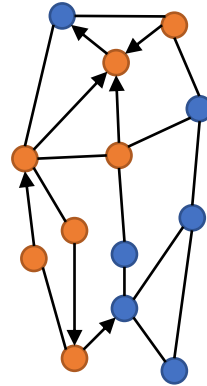
- **Fast drop at beginning**
- **Slow drop in intermediate time**

Compare with RECAP
from previous slide



Conclusion

- Gossip algorithm demonstrated
- OPC UA proved suitable
 - Reliable, high level implementation
 - Authentication (to be implemented)
 - UA Discovery Service for Plug & Play
- Current follow-up work
 - Comparison with merit-order market
 - Beneficial concepts of gossiping (over merit-order)
 - Conditional bidding
 - Meaningful Cross-Commodity
- Future follow-up work
 - Cross-commodity scenario: **electricity** + **heat** (+ **coolant** + else)
 - Automated introducing of new participants



covalion

Thank you!

