# Final report

## 1.1 Project details

| | |
|---|---|
| **Project title** | Dynamic topology data in distribution grids |
| **Project identification (program abbrev. and file)** | 12227 |
| **Name of the programme which has funded the project** | **ForskEL** |
| **Project managing company/institution (name and address)** | **Danmarks Tekniske Universitet (DTU)**<br>**Anker Egelunds Vej 1, Bygning 101A**<br>**2800 Kgs. Lyngby** |
| **Project partners** | **DTU** |
| **CVR** (central business register) | **30 06 09 46** |
| **Date for submission** | **27/02/2017** |

## 1.2 Short description of project objective and results

**English version:**
In order to efficiently provide system services to the distribution grid, aggregators of flexible DER units require dynamic information about the topology of the grid. This information is currently not available to third parties in real time. The project has developed technical concepts for the decentralized real-time exchange and processing of topology data. This includes node discovery, distributed merging of datasets and topology reduction for efficiency reasons. The viability of the fundamental concepts have been proven in a laboratory implementation.

**Danish version:**
Aggregatorer til de fleksible DER enheder vil have brug for dynamisk information om nettets topologi for at kunne tilbyde avancerede systemydelser til forsyningsnettet på en efficient måde. I dag er denne type information ikke tilgængelig i realtid for tredje part. Projektet har udviklet et teknisk koncept for at udveksle og behandle topologidata i realtid. Konceptet inkluderer node discovery, distribueret fusionering af data og topologireduktion for at opnå forbedret effektivitet. De fundamentale koncepter kunne implementeres i flere laboratorieforsøg.

## 1.3 Executive summary

The proliferation of low-cost ubiquitous communication networks and the trend towards embedding microprocessors and communication interfaces into nearly all technical equipment provides an opportunity for building energy infrastructures with a much higher degree of observability than is the case today. This is of particular importance for the evolution of more complex power system services. The entity coordinating the provision of such a service (e.g. an aggregator) will need to know at which point of the grid each flexible unit under its control is connected at a particular point in time, i.e. it needs dynamically updated information about the grid topology within the area of its operation.

Based on the assumption of a much higher degree of integration between ICT infrastructure and power grid, the project aimed to investigate and evaluate ways in which the topology of a distribution grid could be determined in real time for use by third party smart grid services. To achieve this, it would analyze technical issues associated with the transformation of the data collection infrastructure in distribution grids from a single-source, single-client system to a multi-source, multi-client system. Specifically, it would determine how partial information could be extracted and shared, given the large amounts of data involved. Finally, the project aimed to provide a proof of concept regarding the implementability of the above.

The project results consisted of the following:
(1) a mechanism for the decentralized discovery of intelligent devices ("network nodes") which are able to participate in a topology data service,
(2) a merging mechanism which allows for the decentralized and incremental construction of a base topology,
(3) a data representation which allows an efficient implementation of the above mechanisms,
(4) a topology reduction method used to identify a small number of relevant data sources needed for real-time updates,
(5) an implementation of the above in a software framework for deployment in the laboratory, and results from a proof-of-concept test in the laboratory.

At the present state of development, the results can be characterized as TRL 2-3. In the short term, the findings would be most useful as an input to further research activities. DTU has identified ongoing and future projects that will be able to make use of the results.

## 1.4 Project objectives

### *1.4.1 Background*

The physical conditions of energy production and consumption are changing rapidly. A high and growing share of renewable energy production with fluctuating characteristics and decentralized energy production from small photovoltaic (PV) systems change the energy balance of the power system. The rise of more proactive energy consumers ("prosumers"), i.e. end consumers with local production (PV, combined heat and power), controllable demand and storage (residential batteries, electric vehicles) demands coordination of energy resources at a much larger scale. The shift towards electricity as an energy carrier in transportation and space heating leads to a higher utilization of power grids than previously planned. The proliferation of low-cost ubiquitous communication networks and the trend towards embedding microprocessors and communication interfaces into nearly all technical equipment provides an opportunity for building energy infrastructures with a much higher degree of observability than is the case today, if appropriate communication and data processing infrastructures are developed which are able to handle the flood of available data in an efficient manner.

The electric energy sector has evolved from an integrated hierarchical structure to a less centralized setup with multiple actors. Electricity markets allow third party service providers to participate in the operation of the grid by delivering system services which they obtain by controlling a commercial portfolio of energy resources. The main purpose behind the present setup has been the provision of services requested by the TSO such as regulating power. As the power system moves to a higher penetration of renewables, the ability to harvest unused operational flexibility in DER units, including demand response (DR), is seen as one of the key enabling technologies. As long as these services are provided by relatively few DER units, their location in the grid is not important, i.e. an approach similar to "fit and forget" in the early days of wind power can be taken.

### *1.4.2 Problem statement*

As the number of units grows, the impact of the provision of these services on the distribution grid can no longer be neglected. Additional services to the distribution grid must be defined such as peak shaving or voltage control, and their delivery must be coordinated with the delivery of services to the TSO. The efficiency of these services is very much dependent on the location at which they are delivered. This means that an entity coordinating the provision of such a service (e.g. an aggregator) will need to know at which point of the grid each flexible unit under its control is connected at a particular point in time, i.e. it needs dynamically updated information about the grid topology within the area of its operation. This information would be important for all stages of the aggregation process: Service scheduling, service delivery and validation of compliance by the aggregated units.

This need reveals four interrelated shortcomings of the present infrastructure:

(1) Unlike in transmission grids where near-complete automation of substations results in a fully observable grid and EMS applications are able (by way of a topology processor) to request complete and rapidly updated topology information, the situation in distribution grids is very different. Particularly in the low-voltage (LV) and lower medium voltage (MV) networks, voltage and power flow measurements and connectivity information from circuit breakers, disconnectors and fuses is sparse and large parts of distribution grids are not fully observable. This is not a problem for the present mode of grid operation where topology information is primarily used in connection with Outage Management Systems (OMS), i.e. for tracking topology changes due to planned and unplanned maintenance. Because of wider margins for operation, topology information is not generally used for real-time control of the grid. This usage pattern does not depend on fast dynamic topology updates, and manual

editing of the Distribution Management System (DMS) database (for example based on feed-back from field crews) is common.

(2) The number of nodes and interconnection points of distribution grids is vastly greater than that of transmission grids. Therefore, tracking the complete state of a distribution grid at a high time resolution requires handling very large datasets. If the objective of e.g. an aggregator is to use its portfolio to provide voltage control in a limited part of the grid, an efficient way of extracting the relevant information from the overall grid state is required.

(3) Current DMS systems are designed as single-source, single-client systems. All relevant information is collected from the DSO's dedicated data collection infrastructure (SCADA, sub-station automation) and processed in order to be consumed by the DSO. If the full innovative potential of DER and prosumer resources is to be realized, a transformation of the information infrastructure to a multi-source, multi-client system is required. Such a system would be able to augment measurements and state information from the DSO's internal systems with additional data collected by devices owned by third parties (multi-source). It would also provide a straightforward way for relevant third parties to acquire information about the state of the grid in real time, in order to provide grid services (multi-client). As an example, an aggregator of EV charging posts would be able to augment the DSO's state information with voltage measurements at individual PCCs, and in return have access to the connectivity information required to determine which of the units under its control would be able to provide load relief for which feeder. Neither of these information flows is straightforward to establish in today's systems.

### *1.4.3 Project objective*

Based on the above, the project used the following assumptions:
(1) In order to enable the level of innovation required for a successful transformation of the electrical power system, third parties (private as well as commercial actors) must be enabled to develop and provide services to the power system to a much larger degree than today. Becasue innovation cycles in the IT and telecommunications domain are much shorter than those in the power sector, the possibilities for decentralized and partial deployment of solutions must be improved.

(2) Information, including real-time information, about the state of the power system is a major enabler. If (1) is to be achieved, the information and communication systems used to operate the grid must evolve towards more openness, sharing of information and collaboration. This requirement is not unique to the power sector; it merely follows a general trend in society and various other domains of the economy such as transportation and housing.

(3) Several concepts for the production of real time topology data have been proposed in recent years. Most of these concepts are exclusively concerned with algorithms for the calculation or estimation of the data itself. In order to get to a working system, these algorithms would have to be a part of an integrated framework for data exchange between the actors involved: DSO, aggregator, DER owner, metering provider and possibly others.

These points outline the direction of a fundamental and long-term development. Within the narrower scope of the project, the following objectives were chosen:

(1) Investigate and evaluate ways in which the topology of a distribution grid could be determined in real time for use by third party smart grid services.

(2) Analyze technical issues associated with the transformation of the data collection infrastructure in distribution grids from a single-source, single-client system to a multi-source, multi-client system. Specifically, determine how partial information could be extracted and shared, given the large amounts of data involved.

(3) Provide a proof of concept regarding the implementability of the above.

The focus of the project was strictly limited to issues of data acquisition and processing. It did not include regulatory issues, cybersecurity or other related topics. Furthermore, the project did not primarily aim to develop methods and algorithms for topology detection, but tried to use existing research work and determine how this work could be integrated into present and future scenarios for DER/DR aggregation in distribution grids.

### *1.4.4 Project structure*

The project was organized in the following five work packages (WPs):

WP1: Review and categorization of existing methods for topology detection - Literature review of existing centralized and decentralized methods, categorization and evaluation.

WP2 Selection of baseline scenarios for smart grid services - Identification of potential future grid services which depend on topology information. Description of scenarios and use cases based on these services.

WP3 Design of framework for a topology service - Design and implementation of a software framework which allows different algorithms to be tested in a simulation environment, as well as the implementation of the actual algorithms.

WP4 Proof-of-concept (Simulation) - Conduct simulation experiments to evaluate the general viability and performance of the chosen solutions.

WP5 Dissemination - Evaluation and dissemination of results.

### *1.4.5 Project execution*

The project generally followed the planned course, however three factors had a significant influence on the outcome of the project.

(1) WP4 as initially conceived was relying on being able to make use of a cosimulation framework, the development of which was started as part of the RTLabOS project (ForskEL) in 2013/2014. The cosimulation framework, integrating a power system simulator, a communication simulator and a container for executing control software, was to be used for proof-of-concept testing of the developed algorithms. Contrary to expectations, the cosimulation framework did not reach the level of functionality which would have been required for this purpose, within the time available. Also contrary to expectations (as stated in the risk assessment section of the original project proposal), demand from other ongoing projects did not enable resources to be allocated to the completion of the simulation platform.
As a solution, the project chose to perform laboratory experiments instead of simulation work, using DTU's SYSLAB facility. As an advantage, this provided the added realism of conducting a proof-of-concept on a physical system. As a disadvantage, the greater effort of a laboratory implementation vs. simulation work meant that fewer tests could be conducted. Furthermore, no upscaling tests could be conducted due to the fixed size of the laboratory.

(2) The initial analysis work in WP1 revealed less existing work in the area than anticipated. It also revealed that a lot of the existing work did not have a focus on practical applicability. As a result, more research effort needed to be dedicated to finding algorithms from other technical application areas as well as to the adaptation or development of such algorithms, in order to be able to conduct the implementation work required for WP4.

(3) The project shared a PhD student with the EU FP7 ELECTRA project (i.e. the PhD student was conducting work on both projects' resource budgets). The ELECTRA project is partly based on the same assumptions as the Dynamic Topology project, i.e. those of a grid with an embedded data acquisition and distribution infrastructure which is available to multiple interested parties. This overlap was broad enough for some of the work to be useful for both pro-

jects, specifically some of the work on topology reduction (under task 3.1), the design of the framework (under task 3.2) and the testing (under task 3.3 and in WP4).

## 1.5 Project results and dissemination of results

### *1.5.1 Results overview*

Project results consisted of the following
(1) a mechanism for the decentralized discovery of intelligent devices ("network nodes") which are able to participate in a topology data service,
(2) a merging mechanism which allows for the decentralized and incremental construction of a base topology,
(3) a data representation which allows an efficient implementation of the above mechanisms,
(4) a topology reduction method used to identify a small number of relevant data sources needed for real-time updates,
(5) an implementation of the above in a software framework for deployment in the laboratory, and results from a proof-of-concept test in the laboratory.

The following subsections will describe these results in more detail.

### *1.5.2 Discovery mechanism*

The fundamental idea behind distributed topology detection is that, instead of maintaining a single, central topology database, small local pieces of topology data are available throughout the system and can be combined to a full picture without the help of central infrastructure. For example, an intelligent device in a substation would possess a representation of the substation topology, including identifiers of all outgoing or incoming conductors. Likewise, an intelligent device on the premises of a residential or industrial customer could be programmed to know the topology of the network on the premises, behind the point of common coupling.
In order to combine these pieces of information, and in order to make them available locally throughout the network, several processing steps are needed:

1. The distributed intelligent devices need to find relevant neighbors, i.e. those other nodes who possess topology data from the opposite end of any incoming our outcoming conductor (line, cable). The required discovery mechanism is described in this subsection.
2. All neighbor devices which hold partial topology information about grid sections linked by a common electrical conductor, must merge this information in an iterative process in order to obtain a complete system topology at the end. This topology must be made available to all intelligent devices in the network. This merging process requires large and growing datasets to be moved around the network and needs to be organized efficiently in order to be scalable. This merging mechanism is described in subsection 1.5.3, and an efficient data representation format is discussed in subsection 1.5.4.
3. The two previous steps have only established a shared set of static topology data, i.e. the network of all possible connections. In order to be useful for smart grid applications, this dataset needs to be augmented with dynamic data, such as e.g. circuit breaker positions. Frequent updates of the dynamic data are required, whereas the static data changes much less often. Due to the size and complexity of distribution grids, sending all real time data to all nodes is not realistic and does not scale at all. One possibility to solve this problem is to calculate a reduced topology at each node, such that remote parts of the grid are represented in less detail. This makes it possible to discard all sources of dynamic data which do not have an impact on the local dynamic topology, such as e.g. a circuit breaker in a sibling feeder. This allows

the amount of dynamic data to be transferred to be significantly reduced. A method for topology reduction is described in subsection 1.5.5.

One of the simplest methods for discovering communication entities with a known identifier is represented by one of the fundamental building blocks of the TCP/IP stack - the Address Resolution Protocol (ARP). ARP is used at the link layer to find the physical network endpoint identifier (MAC address) for a given IP address. When preparing a data transmission, the IP address of the destination entity is typically either known beforehand or can be retrieved from a directory service (e.g. DNS). Since IP addresses are virtual identifiers and their assignment to communication entities can change, a static and unique address is needed to physically send a packet to the destination.
The ARP protocol is very simple and consists of a broadcast message in which the sender asks all entities within the network domain to return their MAC address if they have been assigned a particular IP address. In the normal case, exactly one entity will respond.

The ARP protocol can be directly applied to the problem of constructing a communication graph of computing nodes which follows the topology of the underlying power grid. We assume that the grid has been partitioned into sections, each of which is represented by one intelligent device or **node**. (An example of such a partitioning is given in the following section). We also assume that each node has access to a precise partial model of the grid topology in the partition under its control, and knows the globally unique identifiers of all outgoing conductors, i.e. lines, cables etc. crossing the partition boundary. (An example of such a model is given in a later section). Analogous to ARP, the node can then broadcast the identifier of each outgoing conductor and ask for the identifiers of the nodes which are connected to the other end of the conductor. This protocol is shown in the top two panels of figure 1.

The method described above has two major issues with respect to practical implementability in a real-world power system.

1. Like other algorithms based on network-wide broadcasts, ARP-derived protocols do not scale well. The bandwidth cost for a full network discovery is of the order $O(n^2)$ which prohibits applications beyond a few hundred nodes. (The use of ARP itself is limited to local-area networks).
2. On the internet, and many other wide-area networks, broadcast messages are not being forwarded. (This is actually related to the first point, i.e. the high bandwidth cost of broadcasting in large networks with many nodes).

These two issues can be mitigated using the following approaches:

A divide-and-conquer scheme can be applied to the simple algorithm in order to address the first issue, scalability. To achieve this, the whole network is subdivided into nested layers of groups and subgroups such that the combined set of nodes in all layer n-1 subgroups of layer n group G is equal to the set of all nodes in G, i.e. each node is a member in exactly one group at each layer. Different ways exist for obtaining the boundaries of the subgroups; a simple method would be a subdivision by geography. In the example shown in figure 1 (left panel in the 2nd row), three group levels have been created, with each level 3 group (yellow) containing two level 2 groups (green) which in turn each contain two level 1 groups (blue).
Multicast group addresses are now being assigned to each group. In the example, seven multicast addresses are needed (4x level 1, 2x level 2 and 1x level 3). Each node knows the three groups it belongs to, and their multicast addresses. The same question-reply protocol as above is being used, but the initial query will only be sent to the innermost (level 1) multicast group. If no reply is received after a timeout period, the search area will be widened by sending the query, this time to the group corresponding to the next higher level and so on. A search request over three levels is shown in the four panels in the middle and bottom rows of figure 1.

The second issue, the inability to send broadcast (and to some extent, multicast) packets over most wide area networks, can be mitigated by introducing an overlay network which supports these packet types. By choosing an efficient structure for the overlay network, the efficiency of the discovery algorithm can be improved.
The design of such an overlay network is outside of the work of this project.

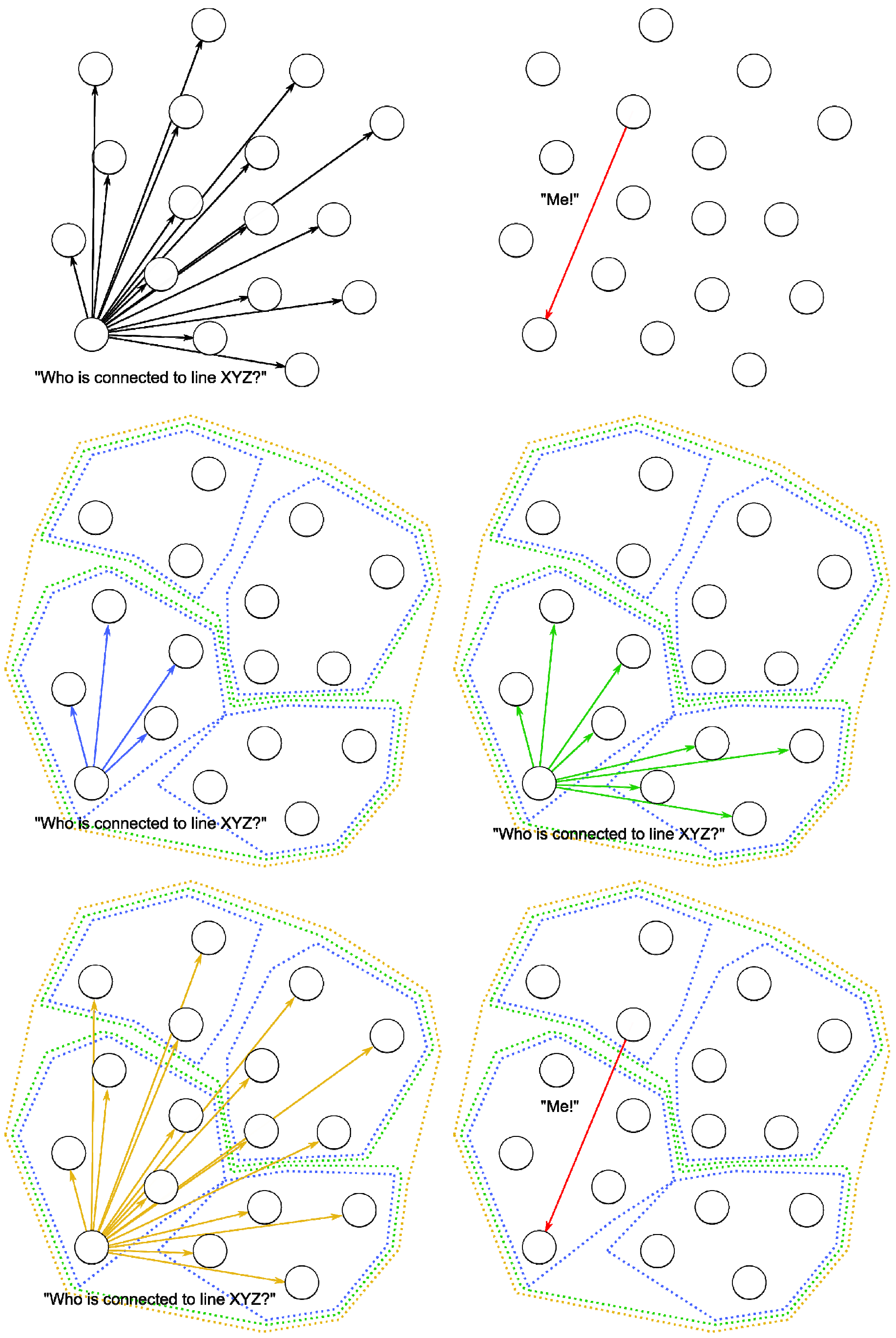

"Who is connected to line XYZ?"
"Me!"
"Who is connected to line XYZ?"
"Who is connected to line XYZ?"
"Who is connected to line XYZ?"
"Me!"

Figure 1: ARP-like protocol (top two panels) and divide-and-conquer extension (middle and bottom panels)

### *1.5.3 Mechanism for merging partial topologies*

The next step is the creation of a static base topology, i.e. a complete set of connectivity data without consideration of dynamic data such as circuit breaker positions etc. Following the overall assumptions made, this base topology should be established without the help of a centralized entity.
For the purpose of discussing the merge algorithms, the example topology on the left side of figure 2 has been created. All components of the power system have been divided into groups according to their geographical and organizational proximity, with each group represented by an intelligent device or **node**. This grouping is reflected by the red dashed boundary lines. In this case, it has been assumed that each substation forms its own group, with all resources within the substation represented by one node. Furthermore, consumers (here shown as loads) are assumed to be equipped with one node each which monitors resources and grid assets behind the point of common coupling (PCC).
Each node has a unique identifier; this is represented by the bold red letters adjacent to the group outlines.
Applying the discovery algorithm described in the previous section yields an undirected cyclic graph of connected nodes which is shown on the right side of figure 2. The graph represents an overlay network which can be used by the nodes to communicate directly with their neighbors in the electrical topology.

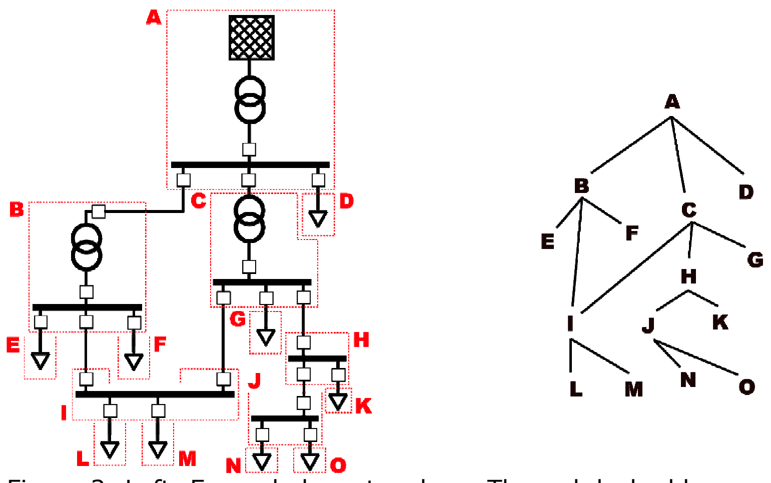


Figure 2: Left: Example base topology. The red dashed boxes indicate which parts of the topology are handled by which network node. Bold letters indicate network node names.
Right: Resulting overlay network graph after discovery.

As a starting point, it is assumed that each node knows the detailed network topology within its boundary. This is a reasonable requirement since the local topology of any site is known locally, and could be programmed into the local node by a utility technician (for substations) or an electrician (for private installations behind the PCC) whenever changes are made to the installation.
The challenge is now to combine all of these bits of local topology information into an overall, coherent static topology of the entire system which is accessible from each individual node. Given the established overlay network on the right side of figure 2, a trivial method would be to start at one randomly chosen node and exchange topology information with all of its neighbors, then continue with the neighbors' neighbors etc. until no nodes are left outside the network. One of the main challenges with this approach is that the merged topology resulting from every exchange of topology information needs to be forwarded to all other nodes before further merge operations can take place. This causes the message cost to grow quadratically with the number of nodes as the merge operation progresses.
Somewhat better performance can be achieved if the merging is allowed to start from multiple nodes instead of a single one. For the description of this kind of greedy algorithm we'll make the following assumptions:

- Each node executes the same algorithm, but it is assumed that the individual nodes start the execution at different points in time. (The algorithm will still work if this assumption is not true; however, in a real-world setup, nodes will not be synchronized unless synchronization is explicitly designed into the system. The assumption will therefore correctly describe most systems).
- Each node has been assigned a globally unique identifier. Proven methods exist for allowing each node to generate its identifier without the need for a central authority, for example based on network MAC address or a random number generator with high entropy.
- Each edge of the graph has been assigned a globally unique identifier (which fulfills the function of an edge weight in a weighted graph). A simple way of generate an identifier for an edge from node X to node Y without the need for a central authority is to calculate a hash function over (a,b) where a and b are the UIDs of X and Y, with b>a. The latter condition ensures that both X and Y generate the same edge identifier for the edge shared by them.
- Each node belongs to a group of nodes (which will grow as the algorithm executes). Initially, each node belongs to a separate group containing only itself. Each group has one leader node which is determined by subsequent elections. Initially, since each group contains only one node, each node is leader of its group.
- Each node has an internal *level* variable which is initially set to zero. The *level* variable is an indicator of the order of the group which the node belongs to.

The greedy algorithm can then be described as follows:

1. The group leader initiates the identification of the outgoing edge with the lowest edge ID leaving the current group. This is done by broadcasting an identification message along the tree. Each node will either pass the message along to its neighbors if it is a member of the group which initiated the message, or it will return the message if it belongs to a different group (=is outside the current group). The edge IDs leading to outside nodes will then be convergecast back to the leader, with each node passing back only the lowest edge ID.
   If no outgoing nodes are available, the algorithm terminates.
2. The leader then sends a merge command to both nodes on the identified edge. If the *level* variables of both nodes are identical, a new group of level n+1 is formed which includes all nodes of both merged groups. All nodes' *level* variables are set to n+1. Leader of the new group will be the node on the merge edge which has the lowest node ID.
   If the *level* variable on both nodes is not identical, the nodes of the group with the lower level will be added to the group with the higher level, and the group with the lower level will cease to exist. All its nodes adopt the *level* of the surviving group.
3. The node on the merge edge which has the lowest node ID merges the partial topologies from both ends of the edge and broadcasts the result along the tree.
4. A new round begins (step 1).

Figure 3 shows the above algorithm being applied to the example graph from figure 2. The top left panel shows the node identifiers (A-O) and randomly assigned edge identifiers (weights, 1-15). For the sake of illustration, only one node starts the execution of the algorithm in each timestep (blue arrows).

At **time t=0**, node B activates and identifies its neighbors A, E, F and I as belonging to different groups. Consequently, B-A, B-E, B-F and B-I are outgoing edges, of which B-E has the lowest edge ID. Nodes B and E both belong to a one-node group of level 0, which causes them to merge into a new group of level 1. B remains the group leader as the node with the (lexically) lowest ID.

At **time t=1**, node D activates and forms a level 1 group with A, following the same mechansm as before. A is the new group leader. At the same time in the group led by B, B-F is identified as the outgoing edge with the lowest ID, and F joins the group. Since F was part of a level 0 group, the level of the enlarged group does not increase, and B remains leader.

At **time t=2**, node O activates and forms a level 1 group with J, led by J. C joins the group led by A, not changing its level or leadership status.

At **time t=3**, node C activates; however, since it has already joined the group led by A, it does not attempt to contact its neighbors on its own (but takes part in the minimum outgoing edge ID search initiated by A). Node I joins the group since C-I is the edge with the lowest ID. The group led by B identifies A-B as the lowest outgoing edge, causing the groups led by A and B to merge. Since both groups were level 1 groups, the new group receives level 2 and is led by A. At the same time, H joins the group led by J.

At **time t=4**, C-H is the outgoing edge with the lowest ID for the groups led by A and J, causing the group led by J to be absorbed into the larger one.

In the following steps **t=5 to t=10**, all remaining single-node groups are absorbed into the large group one by one.

The above example reveals a major weakness of the greedy algorithm. As can easily be seen in figure 3, merging progress is slowed significantly once the initial groups have expanded, leaving only single-node groups to be absorbed. Because of the requirement to broadcast the merged topologies to all members of the group after each merge step, this absorption process must be conducted sequentially, which is expensive in terms of execution time, but particularly in terms of message cost. Furthermore, the need to absorb many single-node groups arises near the end of the merge process, when the size of the merged topologies is already large. This requires a high amount of bandwidth.

It is easy to demonstrate that the most efficient way of conducting the merge process would be by letting groups merge as late as possible, to allow them to grow large before merging. This would parallelize the merging process as much as possible. The simple greedy algorithm does not guarantee this, and can therefore lead to an unbalanced merge where large groups absorb small groups before they can exploit parallelism.

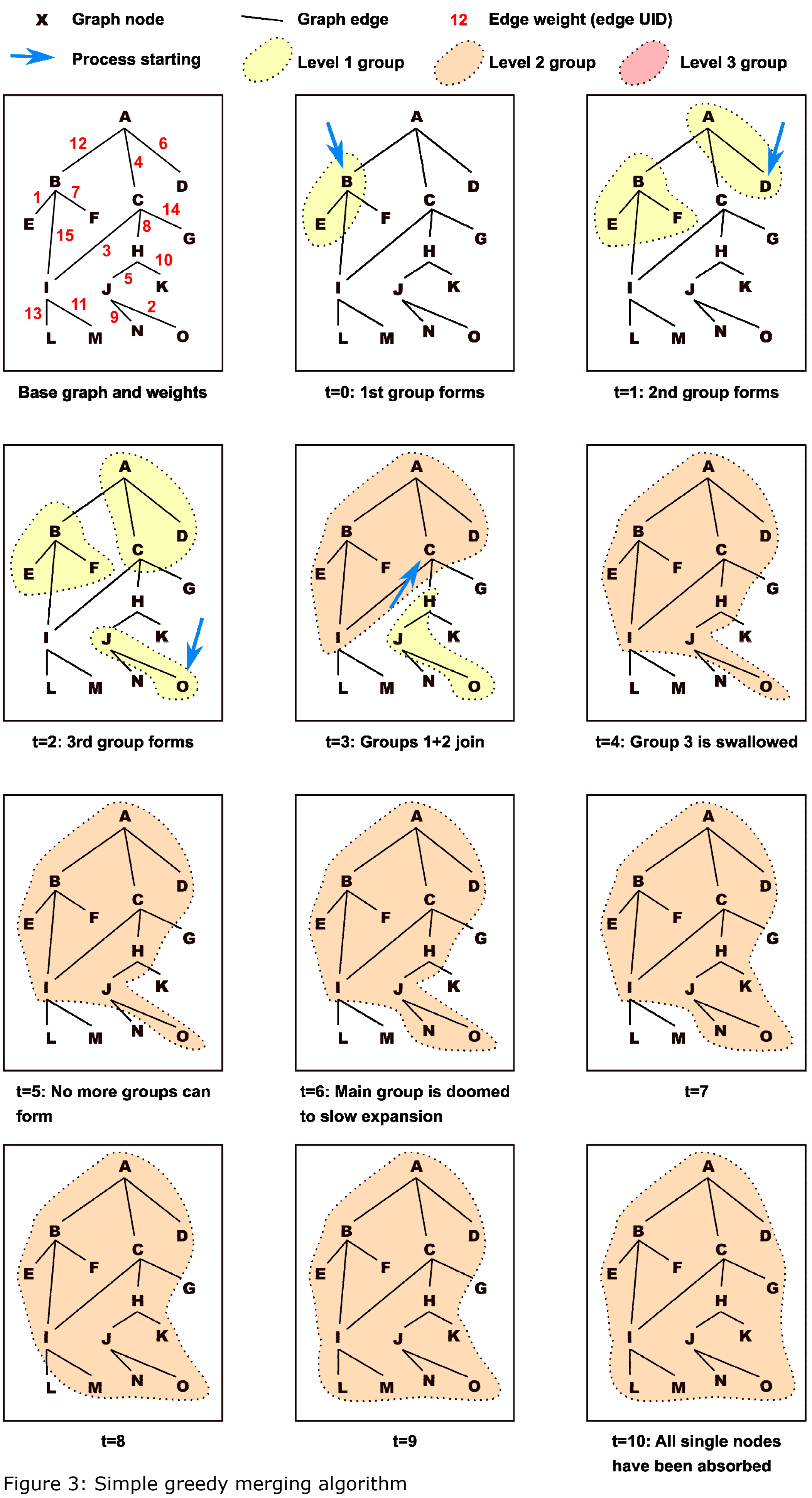


Figure 3: Simple greedy merging algorithm

An improved algorithm has been developed as part of the project. It is inspired by the well-known algorithm for the distributed construction of a minimum spanning tree by Gallagher, Humblet and Spira (GHS). While the problem at hand is less constrained than the latter (no unique spanning tree is required; instead, all interconnections are to be included), it shares the challenge of inefficiencies due to unbalanced growth with the minimum spanning tree problem. The chosen solution uses a similar approach as GHS, introducing additional search and merge constraints based on the relationship between group levels at both ends of a graph edge.

The algorithm uses the same assumptions and the same general working principle as the simple greedy algorithm, but introduces the following additional constraints:

1. During the search phase (step 1 of the greedy algorithm), nodes at the periphery of a group probe the nodes at the opposite end of each outgoing edge. If the outside node's group level is below the inside node's group level, the corresponding edge is removed from the calculation of the minimum edge identifier, unless the outside node does not have any other possibilities than joining the group (i.e. unless the outside node is isolated).
2. During the merge phase (step 2 of the greedy algorithm), the leader will not command a merge along an edge where the group level of the outside node is higher than that of the leader's group, unless no other actions exist for the group (i.e. no other viable outgoing edges exist). This requires the total number of outgoing edges to be counted and reported in parallel to the convergecast of step 1. The leader will instead initiate another search for the lowest ID outgoing edge, this time with the previously chosen edge excluded from the search.

Figure 4 shows the application of the improved algorithm to the same example problem as before. During timesteps **t=0** and **t=1**, both algorithms behave identically. However, at **time t=2**, rule 1 prevents node C from being absorbed by the group led by A, despite A-C being the edge with the lowest identifier. Instead, the level 1 groups led by A and B join into a level 2 group due to B-A being the edge with the lowest identifier leading to a group of equal level.

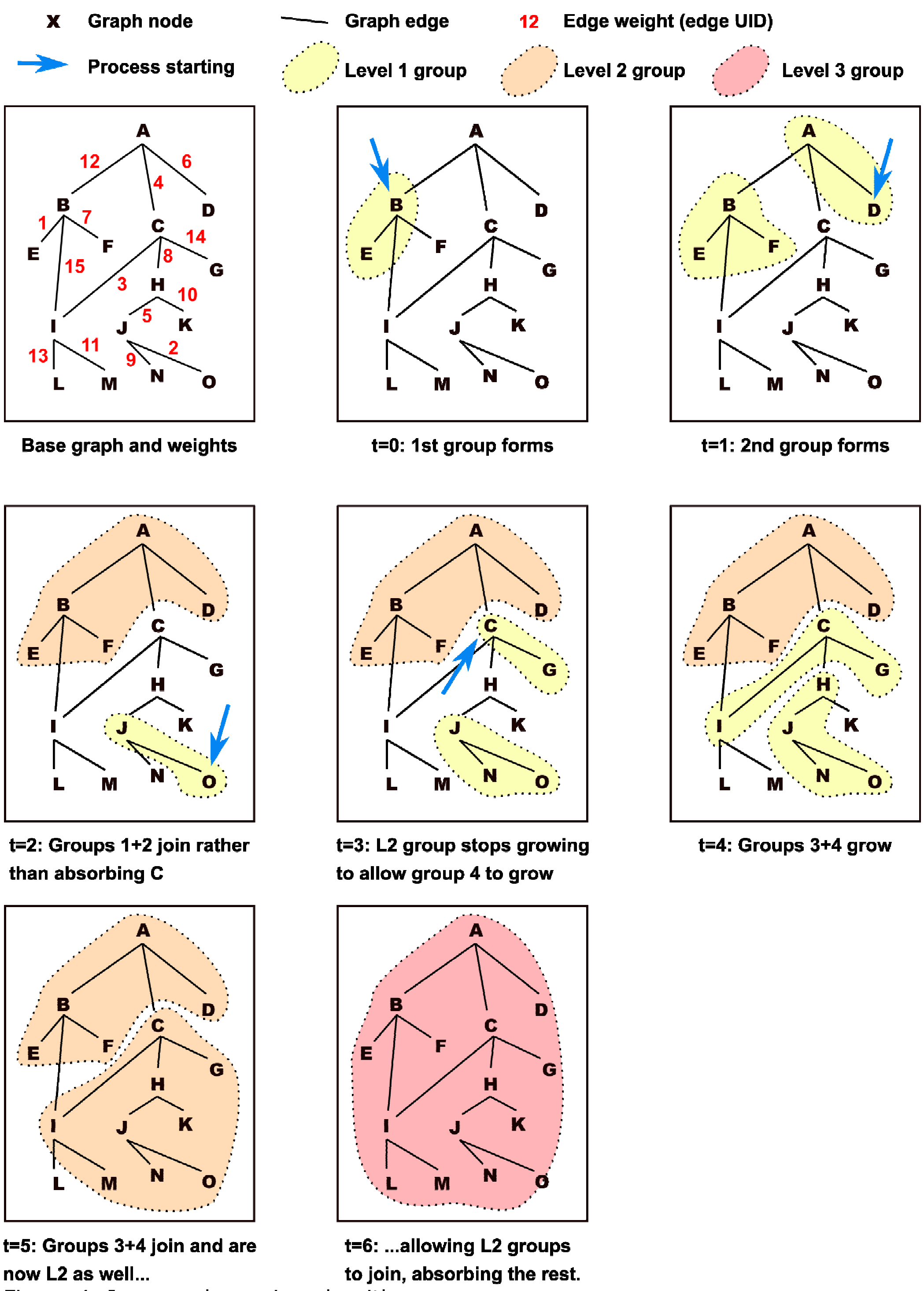
At **time t=3**, the new level 2 group is not able to grow any further without violating rule 1. This allows a new level 1 group consisting of nodes C and G to be formed. The group led by node J first absorbs isolated node N despite J-N having a higher edge ID than J-H. This is due to rule 2; neither node H nor N are at the same level as the J group, but node H still has other unfulfilled merge options towards node K and node C whereas the only possible action for node N is to merge with the J group.

At time **t=4**, the level 2 group led by A is still waiting for other groups to either reach level 2 or run out of other options than merging with the A group. Meanwhile, the two level 1 groups each absorb a level 0 node along the outgoing edge with the lowest ID. In each case, it is the absorbing group rather than the absorbed node which has run out of alternatives.

At **time t=5**, each of the two level 1 groups absorbs one isolated node (K and M, respectively). With the J-led group out of further nodes to absorb, both level 1 groups join into a new level 2 group led by node C.

Finally, **at time t=6**, the new level 2 group absorbs the last isolated node (L). After waiting for several timesteps, the first level 2 group now has a level 2 neighbor. Both level 2 groups join into a level 3 group and the algorithm terminates because no outgoing edges are left.

Despite the seeming paradox of achieving faster termination by halting the growth of one of the groups for several time steps the example illustrates how the improved algorithm achieves better performance by allowing groups to grow - in parallel - to similar size before merging. The performance improvement is in part due to the ability of multiple smaller groups to absorb isolated nodes in parallel, but also because the broadcast and convergecast operations required in each timestep operate on smaller groups until very close to algorithm termination.


X Graph node
Graph edge
12 Edge weight (edge UID)
Process starting
Level 1 group
Level 2 group
Level 3 group
Base graph and weights
t=0: 1st group forms
t=1: 2nd group forms
t=2: Groups 1+2 join rather than absorbing C
t=3: L2 group stops growing to allow group 4 to grow
t=4: Groups 3+4 grow
t=5: Groups 3+4 join and are now L2 as well...
t=6: ...allowing L2 groups to join, absorbing the rest.
Figure 4: Improved merging algorithm

### *1.5.4 Data representation*

Structured message exchange and processing of information requires a meaningful representation of the data under consideration. An information model needs to be designed to supply the applications on top with the required level of information, while keeping the complexity at reasonable levels to avoid overhead.
Two general types of data models are commonly used for representing power system data. One is known as the node-breaker model which is useful for describing static system configurations. The other one is known as the bus-branch model which differs from the node-breaker model primarily by incorporating the dynamic positions of switchgear and protection devices. Effectively, the node-breaker model describes the base topology which includes all physical connection states that can be realized. The bus-branch model on the other hand describes exactly one of these states, which allows it to be less complex.

Figures 5 and 6 demonstrate the difference between the two types of data models.

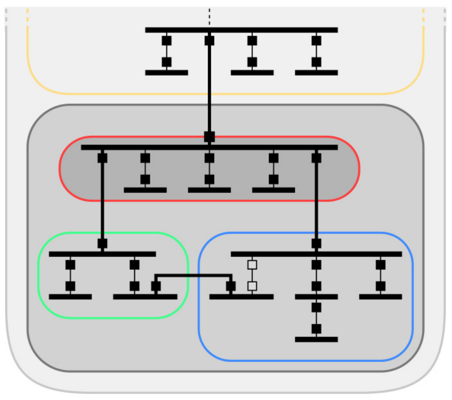


Figure 5: Node-breaker model of an example grid

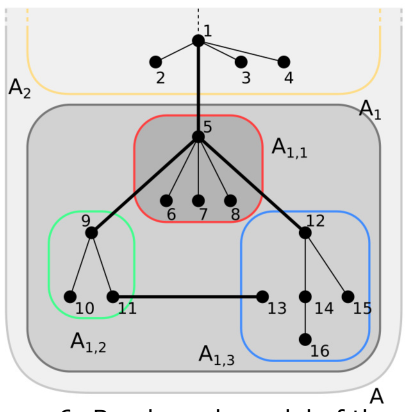


Figure 6: Bus-branch model of the same grid. Not all nodes are connected, reflecting open circuit breakers.

One fundamental design step of distributed information systems is to decide, after a process has been broken down into individual steps, which of these steps are being executed by which entities. For most distributed systems there are multiple options which all fulfill the same purpose, but may have widely different properties, strengths and weaknesses. This is illustrated in figure 7, using design variations of a simple client-server based data retrieval system as an example.

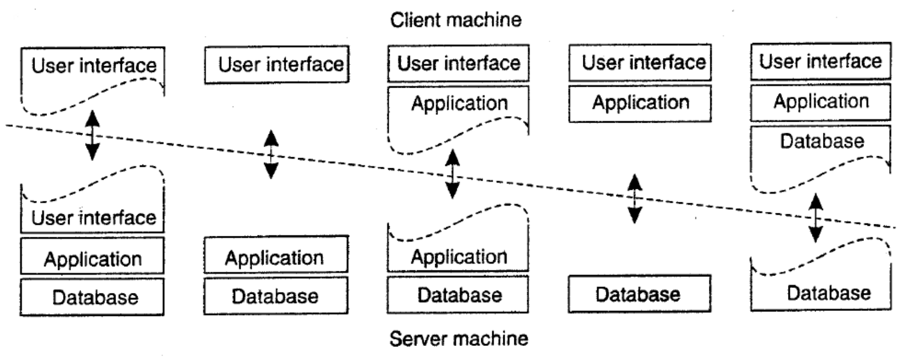


Figure 7: Example: Different ways of splitting a distributed applications between entities. (Source: A. Tanenbaum and M. van Steen - Distributed systems. Pearson, 2006)

It is easy to see that the choice of design has a decisive impact on the type of data that needs to be communicated between systems, and on the requirements towards the communication channel (e.g. reliability, bandwidth etc.).

In our case of a system to determine electrical grid topology in a distributed fashion and from distributed data sources, three relevant steps can be distinguished:

1. Assembly of a global static model from partial models available locally
2. Reduction of the complexity of the global model, leaving only parts of the model relevant to the specific location
3. Acquisition of real-time state information and transformation into a dynamic model

Each of these steps can be performed in different locations, and the steps can be performed in different sequences; both decitions will have a significant impact on the performance of the overall algorithm. The following combinations have been analyzed to be meaningful:

a) First exchange partial node-breaker models between nodes and build a global node-breaker model. Then locally acquire dynamic state information and transform each instance of the node-breaker model into a bus-branch model. Finally, reduce the bus-branch model locally.

b) First exchange partial node-breaker models between nodes and build a global node-breaker model. Then locally reduce each instance of the node-breaker model locally. Finally, acquire dynamic state information needed to transform each reduced node-breaker model into a reduced bus-branch model.

c) First, acquire dynamic state information needed to transform each partial node-breaker model into a partial bus-branch model. Then exchange the partial bus-branch models between nodes and build a global bus-branch model. Finally, reduce the bus-branch model.

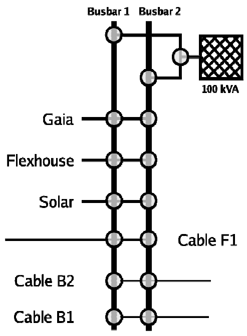
As discussed earlier on, b) is the most attractive of these three possibilities. Option a) requires each node to request a full set of dynamic data for the model transformation of the unreduced model, which is inefficient. Option c) Requires the distributed exchange and assembly of the model to be repeated every time a single circuit breaker changes position, which is inefficient as well.

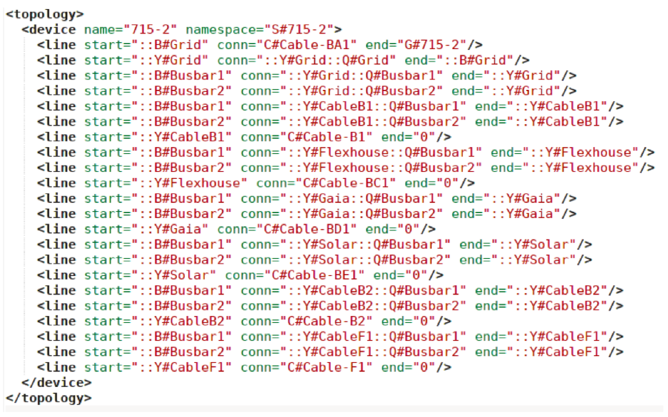
The dominant standard for representing and exchanging grid topology data is the Common Information Model (CIM). CIM is an object model which enables the representation of the node-breaker model of a power system in high detail. However, the high level of detail in CIM models is inevitably linked to a large overhead which is acceptable for its original design purpose of exchanging large and very slowly changing datasets. This however makes CIM less suitable for the real-time exchange of data. While real-time topology data exchange has been proposed as a use case, and CIM variants with a lower overhead have been proposed for this purpose, none of these are currently part of the standard.
For the purpose of this project, a simplified modelling approach has been chosen instead which is based on the CIM model but significantly reduces the amount of information.

The two main simplifications of the chosen model versus the CIM model are:

- Elimination of the redundancy related to ConnectivityNodes
- Reduction of CIM's many equipment classes to four universal types: Ideal conducting equipment (busbars, connectors), nonideal conducting equipment (lines, transformers), dynamic conducting equipment (breakers, switches, plugs) and end resources (generators, loads). Ideal conducting equipment can then be ignored or respectively be treated as a connection point. Nonideal conducting equipment can also be treated as a connection point if only the general connectivity is of interest.
- The addition of open-ended conductors which are a crucial necessity for being able to generate valid, self-contained partial topologies whose neighbors will not be known before the discovery process starts.

The chosen data representation is shown in figure 8, using one part of DTU's SYSLAB laboratory as a simple example. (The entire laboratory grid was mapped to the chosen data representation as part of the project). The top left of figure 8 shows the elctrical single-line diagram of a substation switchboard with two busbars and circuit breakers (circles). At the bottom of the figure, the electrical layout has been mapped to an object model. It can be seen that each bay contains an open-ended connector each of which corresponds to one conductor crossing the boundary of the partial topology represented in this case.





```
<topology>
  <device name="715-2" namespace="S#715-2">
    <line start="::B#Grid" conn="C#Cable-BA1" end="G#715-2"/>
    <line start="::Y#Grid" conn="::Y#Grid::Q#Grid" end="::B#Grid"/>
    <line start="::B#Busbar1" conn="::Y#Grid::Q#Busbar1" end="::Y#Grid"/>
    <line start="::B#Busbar2" conn="::Y#Grid::Q#Busbar2" end="::Y#Grid"/>
    <line start="::B#Busbar1" conn="::Y#CableB1::Q#Busbar1" end="::Y#CableB1"/>
    <line start="::B#Busbar2" conn="::Y#CableB1::Q#Busbar2" end="::Y#CableB1"/>
    <line start="::Y#CableB1" conn="C#Cable-B1" end="0"/>
    <line start="::B#Busbar1" conn="::Y#Flexhouse::Q#Busbar1" end="::Y#Flexhouse"/>
    <line start="::B#Busbar2" conn="::Y#Flexhouse::Q#Busbar2" end="::Y#Flexhouse"/>
    <line start="::Y#Flexhouse" conn="C#Cable-BC1" end="0"/>
    <line start="::B#Busbar1" conn="::Y#Gaia::Q#Busbar1" end="::Y#Gaia"/>
    <line start="::B#Busbar2" conn="::Y#Gaia::Q#Busbar2" end="::Y#Gaia"/>
    <line start="::Y#Gaia" conn="C#Cable-BD1" end="0"/>
    <line start="::B#Busbar1" conn="::Y#Solar::Q#Busbar1" end="::Y#Solar"/>
    <line start="::B#Busbar2" conn="::Y#Solar::Q#Busbar2" end="::Y#Solar"/>
    <line start="::Y#Solar" conn="C#Cable-BE1" end="0"/>
    <line start="::B#Busbar1" conn="::Y#CableB2::Q#Busbar1" end="::Y#CableB2"/>
    <line start="::B#Busbar2" conn="::Y#CableB2::Q#Busbar2" end="::Y#CableB2"/>
    <line start="::Y#CableB2" conn="C#Cable-B2" end="0"/>
    <line start="::B#Busbar1" conn="::Y#CableF1::Q#Busbar1" end="::Y#CableF1"/>
    <line start="::B#Busbar2" conn="::Y#CableF1::Q#Busbar2" end="::Y#CableF1"/>
    <line start="::Y#CableF1" conn="C#Cable-F1" end="0"/>
  </device>
</topology>
```

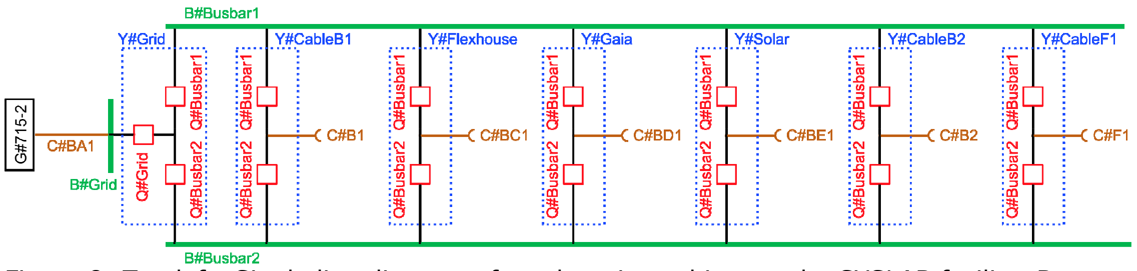


Figure 8: Top left: Single line diagram of a substation cabinet at the SYSLAB facility. Bottom: Corresponding graphical representation using the chosen data model. Top right: Output of XML mapping.

### 1.5.5 Topology reduction method

Exchanging and compiling the static topology information (node-breaker model) as shown in the previous sections is a relatively expensive operation in terms of processing time or message cost, but does not need to be performed very often. A much bigger issue is the real-time compilation of the bus-branch model from the node-breaker model and breaker state information. This needs to be done frequently in order to be useful, but sending all breaker positions in the whole grid to all nodes is very inefficient ($n^2$). If it could be identified which parts of the topology are of interest at a particular location, individual breaker positions could be selectively polled.

Reduction techniques are generally employed whenever the complexity of models is either too high for their intended purpose, or it would render the targeted problem infeasible. Some existing works reduce the dynamic model of power system components while retaining the dynamic response and number of inputs and outputs. Other works concern themselves with the topological reduction of printed ciruit boards, in cases where only the low frequency characteristics of the circuit are of interest. Common approaches for reducing topological complexity in the field of power engineering are Kron reduction, Ward external equivalents, Dimo's method, and Zhukov's method. They are widely applied for various types of power system analyses of large, interconnected areas. While each of these methods serves a different purpose, they all focus on the electrical aspects of grids but do not take logical groups into account.

The following approach for reducing and assembling topology and binary connectivity information for a power grid retains the logical grouping of buses into predefined areas. It also preserves information about the overall hierarchy of the grid, such that the reduced topology of each area contains complete information of the grid layout inside the area but progressively decreases the level of detail of grid sections at other levels of the hierarchy. Physical connectivity information between areas remains intact.
By using a global naming scheme, the computation of reduced area representations as well as the assembly of the reduced global topology can be performed in a decentralized manner.

Starting point for all further considerations is the oriented graph $G=(V,E)$, with $V$ being vertices/nodes and $E$ edges, and $n=\varkappa(V), m=\varkappa(E)$ as their respective cardinalities. The incidence matrix of $G$ is an $n\times m$ matrix defined as

$$I_{i,j}=\left\{\begin{matrix} -1: v_i \mathit{is the source of}\ e_j \\ 0: v_i \mathit{is the sink of}\ e_j \\ 1: v_i, e_j\ \mathit{are not incident} \end{matrix}\right\} \quad (1)$$

with $v_i \in V$ and $e_j \in E$.

The graph $T_{full}=(B_{full}, L_{full})$ represents the global distribution grid topology, where $B_{full}$ is the set of vertices representing buses, $L_{full}$ is the set of edges representing closed lines between buses, and $n_{full}=\varkappa(B_{full}), m_{full}=\varkappa(L_{full})$ are the respective element numbers. The dimension of the incidence matrix calculates to:

$$Dim(I_{full})=n_{full}\times m_{full} \quad (2)$$

In order to give a reference frame in which the topology reduction and assembly process is conducted, a hierarchical ordering of the grid topology into defined areas $A$ is assumed. The hierarchical level $l$ is determined by the number of subindices. For example, $A$ is the set of areas on the topmost level $l=0$, $A_i$ is the set of areas on the first level $l=1$ for one area $i$, and so on. We call $A_i$ a subarea of $A = A_1, A_2, \dots$, $A_{i,j}$ a subarea of $A_i = A_{i,1}, A_{i,2}, \dots$, i.e. $A \supseteq Ai \supseteq Ai, j$.

For the sake of simplicity, we focus only on two levels and assume that each of the $r = \varkappa(A)$ areas $A_i$ has $r_i = \varkappa(A_i)$ areas $A_{i,j}$, and $r_{i,j} = \varkappa(A_{i,j}) = \Box$.

To the hierarchy level of $A_{i,j}$ belongs the subset $B_{i,j}$ of the complete set of buses $B_{full}$, and the smallest possible area corresponds to a single bus node. Each bus $b \in B_{full}$ belongs to one and only one area $A_{i,j}$, hence $B_{i,j} \cap B_{i,k} = \Box$ and $\cup_{i,j} B_{i,j} = B_{full}$ for $\forall i \in 1 \dots r$ and $\forall j, k \in 1 \dots r_i$, $j \neq k$. The hierarchy level of $A_{i,j}$ also contains domestic lines $L_{i,j}$, which connect the buses $B_{i,j}$, and the tie-lines $L_{tie,i,j}$ each belonging to just one bus in $B_{i,j}$ because their respective other ends cross the area boundaries. Concordantly, $A_i$ has the domestic lines $L_i$ connecting the areas $A_{i,j}$ and the tie-lines $L_{tie,i}$, and $A$ features the domestic lines $L$ and tie-lines $L_{tie}$.

The last set contains all tie-lines that leave the observed grid and is empty for a closed topology. The corresponding graphs of each area on all levels contains the according buses/subareas and domestic lines, e.g., $T_{i,j} = (B_{i,j}, L_{i,j})$.

As a starting point for topology reduction, it is assumed that each area and subarea has an autonomously operating reduction processor, i.e. a separate copy of a software process which acts on behalf of the area or subarea. It is further assumed that all software processes are identical, but receive different input data according to their location in the grid.

Using the grid model given in the previous section, the reduction is then performed locally and entirely decentralized by all areas simultaneously. The reduction process starts at the bottommost level, where the set of connected topology segments $C_{i,j}$ of area $A_{i,j}$ is determined. Each connected segment $c \in C_{i,j}$ is disjoint with the other segments within the area, i.e. there are no connecting lines between them, hence $c_a \cap c_b = \{\},\ \forall a, b \in \{1 \dots r^c_{i,j}\}, a \neq b$, and $r^c_{i,j} = \varkappa(C_{i,j})$. The reduced area $A^{red}_{i,j}$ is formed by a new set of equivalent buses $B^{equ}_{i,j}$, each corresponding to one $c \in C_{i,j}$, and the tie-lines $L_{tie,i,j}$.

Thus, $A^{red}_{i,j}$ maintains the full connectivity information between its tie-lines $L_{tie,i,j}$ without exposing its inner structure. The reduced topologies of all areas $\cup_j A^{red}_{i,j}$ are subsequently propagated upwards in the hierarchy to $A_i$ and are then reduced to $A^{red}_i$ with $B^{equ}_i$ and $L_{tie,i}$, accordingly. The process finishes at the topmost level after reducing $\cup_j A^{red}_i$ into $A^{red}$ with $B^{equ}$ and $L_{tie}$.

The reduced global graph $T^{red}_{i,j} = (B^{red}_{i,j}, L^{red}_{i,j})$ as viewed from (sub)area $A_{i,j}$ is then obtained by collecting and assembling all the topological information from the different levels across the hierarchy, i.e.:

1. The complete local topology;
2. The reduced topology of the areas $Ai, k,\ \forall k 6= j$ within hierarchy level 2;
3. The reduced topology of the areas $A_k,\ \forall k \neq i$ within hierarchy level 1.

The resulting reduced global topology $A^{glob}_{i,j}$ for area $A_{i,j}$ can therefore be expressed as

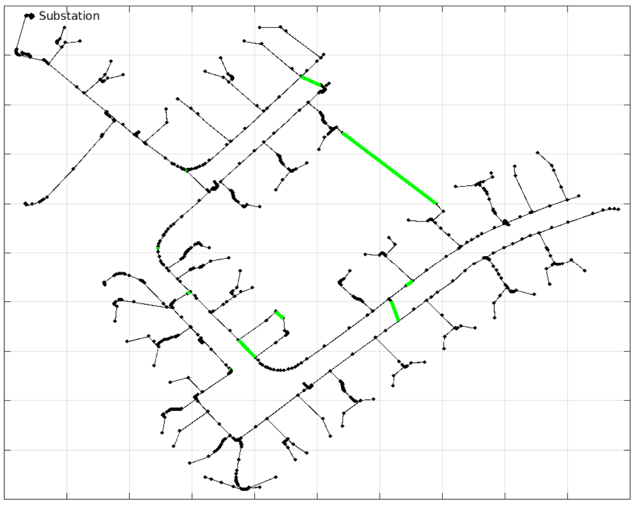
$$B_{i,j}^{\text{red}}=B_{i,j}\cup\left(\bigcup_{k\neq j}B_{i,k}^{\text{equ}}\right)\cup\left(\bigcup_{k\neq i}B_{k}^{\text{equ}}\right)$$
$$L_{i,j}^{\text{red}}=L_{i,j}\cup L_{tie,i,j}\cup\left(\bigcup_{k\neq i}(L_k\cup L_{tie,k})\right)\cup L, \tag{3}$$

with the cardinalities $n_{i,j}^{\text{red}}=\varkappa(B_{i,j}^{\text{red}})$ , $n_{i,j}=\varkappa(B_{i,j})$ , $m_{i,j}^{\text{red}}=\varkappa(L_{i,j}^{\text{red}})$ and $m_{i,j}=\varkappa(L_{i,j})$ . The total number of equivalent buses and lines is therefore $n_{i,j}^{\text{equ}}=n_{i,j}^{\text{red}}-n_{i,j}$ and $m_{i,j}^{\text{equ}}=m_{i,j}^{\text{red}}-m_{i,j}$ . These numbers are finally used to calculate the dimensions of the incidence matrix $I_{i,j}^{\text{red}}$ of the reduced global topology for area $A_{i,j}$ :

$$Dim(I_{\text{red},i,j})=(n_{i,j}+n_{i,j}^{\text{equ}})\times(m_{i,j}+m_{i,j}^{\text{equ}}) \tag{4}$$

For numerical investigations of the matrix in the following section, rows and columns are split into original and equivalent buses and lines. Assuming $¿,j,\ni,j¿n$ and $m_{i,j},m_{i,j}^{\text{equ}}\ll m$ for typical power grids, the reduction gain is apparent.

We will use the feeder shown in figure 9 as an example. It is derived from the IEEE European Low-Voltage Test Feeder which is a 906 bus radial distribution grid connecting to the medium-voltage system through a substation transformer. For studying the behavior of the reduction algorithm in meshed grids, new lines were added to the original topology. Also, some existing lines were removed in order to break areas into disconnected grid segments. The modifications to the feeder have been overlaid onto figure 9 in green color.



For investigating the topology reduction on any topology, areas are assigned automatically by applying Algorithm 1 which randomly splits the input topology into an arbitrary number of areas $r$ . A radial topology must be provided to the algorithm, but also meshed topologies can be supplied by taking a spanning tree and augmenting it with the remaining lines after performing the random partitioning.

The algorithm creates areas with a certain minimal number of buses $n_{min}$ depending on the total number of buses $n$ and areas $r$ in order to achieve balanced area sizes. The choice of $n_{min}=ceil(\frac{n}{2r})$ ensures that the algorithm will always terminate for $1<r<floor(\frac{n}{2})$ . Here, $ceil()$ and $floor()$ return the next higher respectively lower integers of the real-valued in-

put arguments. Recursive application on the resulting areas allows for hierarchical splitting of the topology into the desired number of hierarchy levels.

The algorithm works as follows. Iteratively a random line in the input topology is randomly selected and removed from the set of lines, which necessarily splits the resulting graph into separated areas because of the original topology's radial characteristic. Determining the sets of connected buses that form these areas is done by the function ConnectedAreas ( $T$ ). The number of buses in each area is then counted, and if the number of buses in one of the areas is lower than $n_{min}$ , the selected line is discarded and a new one is drawn. Otherwise, the line is added to the set of tie-lines and a new iteration starts. The algorithm terminates when the number of desired areas $r$ is reached.

```
**Require:** Radial topology graph $T=(B,L)$
**Require:** Desired number of areas $1<r<floor(\frac{n}{2})$
**function** CreateRandomAreas $(T,r)$
        $n \leftarrow \varkappa(B)$
        $n_{min} \leftarrow ceil(\frac{n}{2}r)$
        $L_{sel} \leftarrow L$
        $L_{tie} \leftarrow □$
        $A \leftarrow □$
        **repeat**
                **repeat**
                        $l_{cut} \leftarrow rand(L_{sel})$
                        $L_{temp} \leftarrow L_{sel} □ l_{cut}$
                        $T_{temp} \leftarrow (B, L_{temp})$
                        $C \leftarrow$ ConnectedAreas $(T_{temp})$
                **until** $\varkappa(c) \geq nmin, \forall c \in C$
                $L_{sel} \leftarrow L_{temp}$
                $L_{tie} \leftarrow L_{tie} \cup l_{cut}$
        **until** $\varkappa(C)=r$
        **for all** $c \in C$ do
                $A \leftarrow A \cup (c, L)$
        **end for**
        **return** $A$
end function
```

Applying the random partitioning algorithm to the example grid yields the set of grid areas shown in figure 10. The algorithm generates two hierarchy levels such that each area is further divided into subareas. For the sake of clarity, figure 10 only shows the subareas for area 1 (different shades of purple).

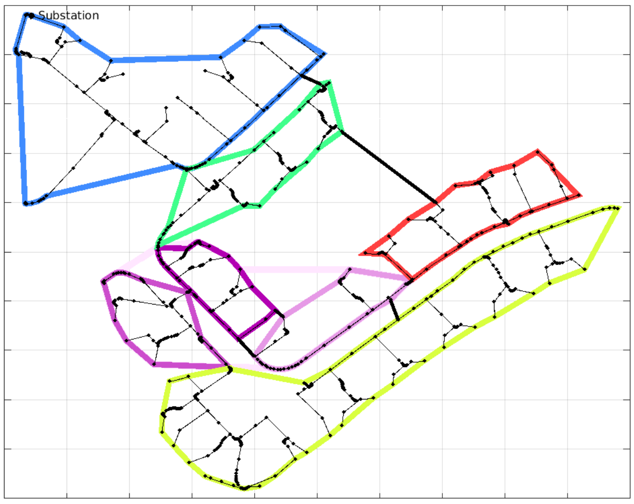


Figure 10: Feeder areas generated by the random partitioning

The reduction levels from the local perspective of subarea 1 is shown in Figure 11. The dark gray patch marks the area of full topological resolution. Mid gray indicates the first reduction level that comprises all other subareas within area 1. The light gray patch includes the reduced topologies of all other areas but not subareas.

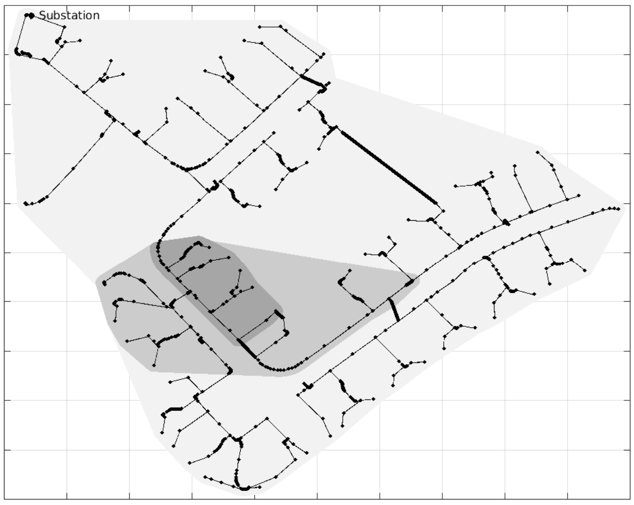
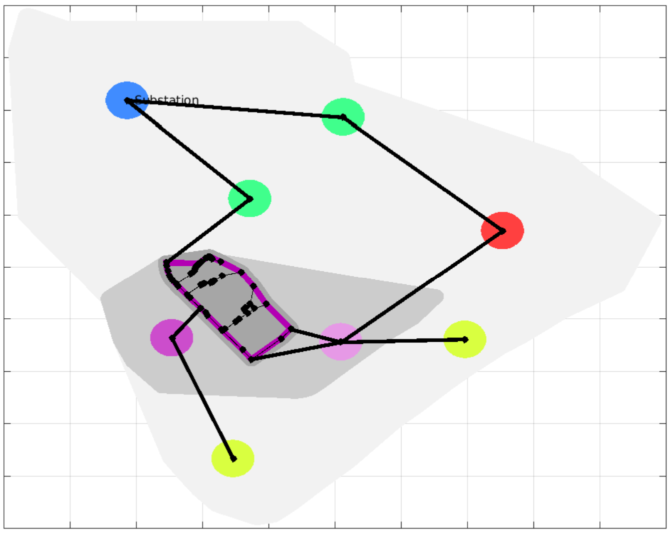


Figure 11: Target reduction levels for other areas and subareas,

Generating the reduced global topology for subarea 1 works as follows. In a first step, all subareas locally perform the topology reduction as described above. The next higher hierarchy levels for each subarea assembles these reduced local topologies (only considering step 2) and performs the reduction in turn. Subarea 1 then collects the reduced topologies of the subareas in area 1 as well as the reduced topologies of all other areas and merges them together. As tie-lines between areas are kept, the resulting topology features the complete mesh information with gradually decreasing resolution for the areas.



Considering only the top-level areas in figure 10, one could assume that area 1 and 4 form a loop over their two tie-lines, as well as area 1, 3 and 5 over theirs. The resulting global reduced topology in figure 12, however, shows that area 3 and 4 are internally split into two disconnected grid segments. The former assumption with area 4 is therefore wrong, and the latter assumption about the second loop is only true because it closes via area 2. Assembling the full local and other reduced incidence matrices as described successfully preserves these internal topology characteristics while keeping the total incidence matrix complexity low. The reduced matrix is 0.067 % the size of the full matrix for a subarea size of 66 buses, which is the expected order of magnitude.

### *1.5.6 Laboratory test*

Three laboratory experiments were conducted in order to test the performance of the different methods separately. All tests were conducted at the SYSLAB facility at DTU's Risø campus.

The SYSLAB facility consists of a 400V, 3-phase grid with a total of 16 busbars in eight substation switchboards on four sites. The four sites are geographically distributed, with dual cable runs of about 250-700m between them. A variety of different DER units - renewable and conventional generators, loads and battery storage units - can be connected to the grid at the different locations. A central crossbar switch allows for flexible changes in grid topology between the different sites. All equipment on the grid is automated and remote-controllable. Each unit is supervised locally by a dedicated node computer which acts as a communication gateway. The node computers run a custom, Java-based software platform which provides communication between nodes, enables deployment of local controllers and logs unit data.

In order to test the performance of the method with different types of grid topology, each experiment was conducted using two different grid topologies, a "long" and a "wide" one. The long topology (figure 13) emulates a weak feeder in the countryside - a long line with several segments and DER units connected to the segments along the line. The wide topology (figure 14) emulates an urban feeder with relatively short line lengths and many branching points.

The lower halves of figures 13 and 14 show how each test topology is mapped onto the laboratory grid, using the flexibility provided by the three-busbar "crossbar" switchboard (center of figure).  In each schematic, filled circles represent closed circuit breakers while outlined circles represent open ones. For the long topology, after "wrapping" through all substations, starting from the grid connection, the line ends at a load simulator after about 3km of cable length.

The wide topology branches out from the grid connection (Cable A1), then again from crossbar A. Each branch has one or several DER units connected to it: A wind turbine ("Gaia"), a building with controllable load ("Flexhouse"), a PV array ("Solar"), load simulators ("Dumpload") as well as storage systems ("Battery").

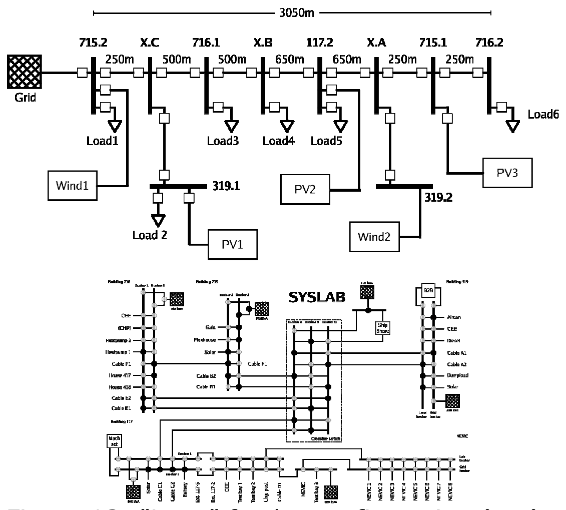


Figure 13: "Long" feeder configuration (top) and representation in the laboratory (bottom)

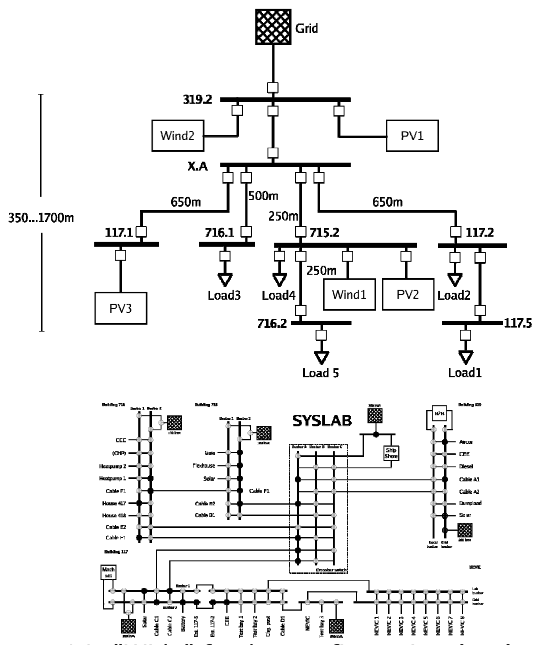


Figure 14: "Wide" feeder configuration (top) and representation in the laboratory (bottom)

The following three experiments were conducted with the laboratory in the "long" configuration. The third experiment was additionally conducted in the "wide" configuration. (The first two experiments do not make use of citcuit breaker position data; the results would therefore be expected to be identical between configurations).

(1) Test of the discovery mechanism. The discovery algorithm is installed on 24 participating nodes (6 substation nodes and 18 nodes representing energy resources) which share a single communication network. A remote-start execution container wraps the algorithm such that a start command can be sent from a remote user interface computer to all 24 instances near-simultaneously (via UDP broadcast). All messages and state changes are timestamped and logged on each machine while the algorithm executes. After the experiment, the 24 logfiles are collated and the execution speed of the algorithm is evaluated.
Figure 15 shows the progression of the discovery process expressed as number of units discovered over time, averaged over 10 runs of the experiment. Despite the small amount of randomness with respect to the starting order introduced by the start mechanism, the algorithm consistently converges to the same result.

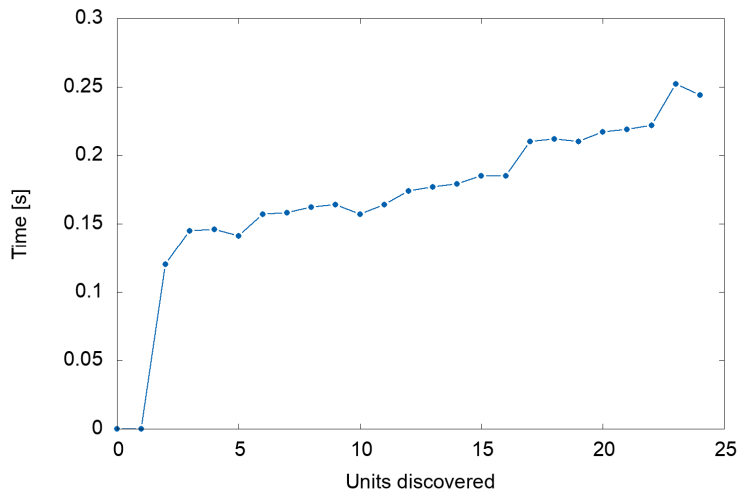


Figure 15: Progress of the discovery algorithm (24 units, average of 10 rounds)

(2) Test of the merging algorithm. The merging algorithm is installed on the same 24 nodes as in (1) and connected to the same remote-start facility. Evaluation is likewise performed by distributed logging of messages and logfile collation.
The progression of the merging process is shown in figure 16, expressed as the size of the largest contiguous merge result in circulation versus time and likewise averaged over 10 rounds. It can be seen that the size moves in discrete steps as the largest component waits for smaller components to merge and eventually reach the same level, at which point a merge operation is initiated. The fact that the steps are clearly discernible even in the averaged results over 10 executions points to the conclusion that the algorithm performs in a stable manner despite the randomness in the starting order of the individual instances, caused by the imperfections of the remote starting mechanism.

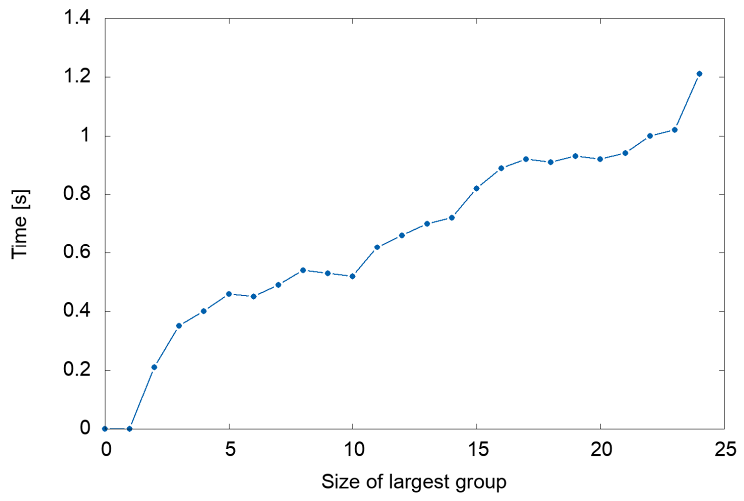


Figure 16: Progress of the merge algorithm (24 units, average of 10 rounds)

(3) Test of the topology reduction method. The topology reduction algorithm is installed on all 24 nodes and given the result of the merging test as an input. Upon synchronized start via the remote start mechanism, each node runs the reduction algorithm to determine the list of relevant circuit breakers needed to obtain the current connectivity. Each node then requests breaker information from the relevant substation nodes and calculates the connectivity matrix of the reduced system.

This test has been executed as a pure feasibility test. No performance measurements were conducted, because differences in node hardware lead to different execution speeds of the algorithm between the nodes.

The primary objective of all experiments has been to prove the implementability and correct function of the individual algorithms in a physical system in a qualitative rather than a quantitative way. The main reason for this limitation has been the limited availability of resources within the project. Going forward, interesting parameters for a quantitative analysis would be (a) the scalability of the individual algorithms as well as their combination, (b) fault tolerance and (c) the robustness of the algorithms to instiantiation order and jitter, i.e. how much the result depends on which instances get to run first.

### *1.5.7 Publications*

A. Prostejovsky, O. Gehrke, M. Marinelli, M. Uslar: "Reduction of Topological Connectivity Information in Electric Power Grids", Proceedings of the 51st IEEE International Universities Power Engineering Conference. Coimbra, Portugal, September 2016.

O. Gehrke, A. Prostejovsky: "An efficient merging algorithm for distributed grid topology information", 8th IEEE International Conference on Smart Grid Communications 2017 (submitted, in review)

### *1.5.8 Fulfillment of objectives*

The project has fulfilled the three objectives stated in section 1.4. The results presented in the sections about discovery and distributed merging have addressed objective (1) which required the investigation of methods for real-time distributed topology detection from distributed data sources.
Objective (2), an analysis of technical issues associated with the development of a multi-source, multi-client system, has been addressed by the results about data modelling and topology reduction.
Finally, the implementation of the laboratory framework and the execution of laboratory tests has addressed objective (3) which called for a proof-of-concept test of the implementability of the results.

## 1.6 Utilization of project results

The current state of the project results can be best characterized as belong to Technology Readiness Levels (TRL) 2 and 3, i.e. some parts of the outlined technologies exist as a solid concept while for others, a proof-of-concept has been conducted, albeit only at the laboratory level. This is not unexpected given the project's exploratory nature.
The project has based its work on the assumption of a different operating model for electrical distribution grids which includes a relationship between system operator, commercial service providers and private households which is quite different from today. Most importantly, it rests on the assumption of a higher degree of data sharing between the different actors, which will not be fulfilled unless political and regulatory pressure causes system operators to embrace such a model. In today's environment, system operators do not have an incentive to do so.

The project results are best seen as an exploration of the technological possibilities which may arise from a different approach to distribution grid operation, something which does not appear to be around the corner. However, while a deployment of the investigated technologies in real-world power grids is unlikely to occur in the near future, the results are directly usable as an input to the ongoing research on future smart grids.

Real-time information about system topology is a key enabler for most localized grid services. Therefore, the significant number of ongoing research efforts investigating these services and the associated market mechanisms could profit from the results of this project. Some of these connections have already been made; as an example, the EU FP7 project ELECTRA collaborated with this project on topology reduction because of a specific concern about so-called "loop flows" which could result from ELECTRA's novel concept for grid operation.
A direct transfer of results is also expected towards the ForskEL project "EnergyCollective" which begins its work in March 2017. The project aims to develop a distributed market for the exchange of residential-level energy services in the lower branches of the distribution system. The decentralized and dynamic acquisition of grid topology data will have a direct application in this context.
Finally, some of the implementation difficulties ahead of the laboratory tests have inspired DTU to start the development of a generalized execution environment for distributed algorithms in a power system context. This is likely to facilitate teaching, smart grid-related laboratory work and possibly future field experiments.

In a longer perspective, this project is part of the ongoing effort to develop the next generation of smart energy systems with a higher degree of embedded automation which are crucially required to enable the green transformation of the energy system.

## 1.7 Project conclusion and perspective

The main conclusions of the project are:

- It is feasible, at least on a limited scale, to supplement today's centralized SCADA systems and topology processors with intelligent infrastructure embedded in the grid, in order to support third-party services and the local availability of grid state.
- Topology acquisition and tracking is possible in a fully decentralized setup, in which none of the local actors initially knows more than the local topology.
- Much of the "heavy lifting", i.e. processor and bandwidth intensive operations, only have to be performed occasionally, when the base topology changes. Adding dynamic data is a comparatively lightweight operation with considerable potential for optimization, in particular by intelligent reduction of the data set.

Several aspects of the work could not be analyzed within the project due to the small size of the project. These would be obvious starting points for future work and include

- a quantitative proof-of-concept of the scalability of the proposed algorithms. Due to the limited size of the laboratory, this would likely require simulation work.
- an analysis of the fault tolerance of the algorithms, and the development of possible mitigation strategies, if required.
- As disccused in section 1.5.2, all of the discussed algorithms would require some form of overlay network to be run upon. While it is easy to create such an overlay network manually, this would not be efficient in larger power systems. An investigation into how to design a suitable overlay network would be a relevant research direction.

As mentioned above, it is expected that some of the results of this project will be found useful in the upcoming EnergyCollective project. The field tests planned in that project may offer an opportunity for further development and testing of the project results.