MIRACLE

Micro-Request-Based Aggregation, Forecasting and Scheduling of Energy Demand, Supply and Distribution

Specific Targeted Research Project: 248195

D3.1 State-of-the-art report on data collection and analysis

Work package 3

Lead partner: AAU

June, 2010

Version 1.0

DOCUMENT INFORMATION	
ID	D3.1 State-of-the-art report on data collection and analysis
Work Package(s)	WP3
Type	Report
Dissemination	Public
Version	Version 1.0
Date	July 5, 2010
Author(s)	Laurynas Šikšnys, AAU; Matthias Böhm, TUD; Torben B. Pedersen, AAU; Christian S. Jensen, AAU; Dalia Martišiūtė, AAU.
Reviewer(s)	Henrike Berthold, SAP

Table of Contents

1 Summary

Today, many countries aim to increase the share of energy consumed that comes from renewable sources. Unfortunately, the electrical power produced from weather-dependent renewable energy sources (RESs; e.g., wind turbines, solar panels) is produced in varying quantities that do not match the varying energy needs. As more and more such renewable energy becomes available, it becomes an increasingly difficult challenge to maintain an energy system that enables the effective use of all available renewable energy. Consequently, tackling this problem is one of the top goals in the energy domain.

The MIRACLE (Micro-Request-Based Aggregation, Forecasting and Scheduling of Energy Demand, Supply and Distribution) project aims to invent and prototype key elements of an energy system that is better able to accommodate large volume of electricity from renewable sources. The approach, taken is based on micro-requests that allow an individual consumer/producer to specify acceptable flexibilities in the amounts of energy consumed and the times when this is done. The introduction of such micro-requests from millions of consumers/producers enables fine-grained scheduling of consumption and production of electricity while maintaining a system-wide balance between demand and supply. In order to appropriately manage very large volumes of micro-requests, a reliable, distributed, and highly scalable computer system infrastructure is needed.

This deliverable concerns Work Package 3, “State-of-the-art report on data collection and analysis” in the MIRACLE project. We first introduce MIRACLE’s application scenario along with the consequent requirements to the data management infrastructure. Then we survey the state-of-the-art of relevant, existing work on data collection, data integration, query processing, and query optimization from the perspective of the projects requirements. Specifically, the survey covers the following key topics: (1) virtually and materialized integrated systems, including column stores; (2) data exchange solutions, including ETL tools, EAI servers, and data stream management systems; (3) web-scale data management; (4) management of uncertainty in the context of probabilistic databases, OLAP, and data streams; (5) management of multi-version data; (6) efficient tracking of continuous processes; and (7) query optimization based on early aggregation and materialized views. Moreover, relevant existing computer systems in the energy domain are covered. For all technologies surveyed, the relation to MIRACLE is discussed.

2 Introduction

The European electricity network has evolved over more than a hundred years, but significant new challenges are still ahead. The need for a liberalized electricity market, better security of electricity supply, and protection of the environment are pushing changes in the European electricity infrastructure [Sas06] today.

Renewable energy plays a key role in producing local, clean, and sustainable energy to meet the growing demand for electricity [Age02]. There are little or no fuel costs associated with the generation of electricity from renewable energy sources (RES - wind, waves, tides, sun). Therefore, many countries try to increase their share in the renewable energy thus reducing the greenhouse gas emissions of the electrical power generation [MWGW10]. In order to accommodate and efficiently utilize the increased amount of renewable energy sources, the countries need to further evolve their electricity networks [Ene09].

The MIRACLE project is a step towards enabling more efficient use of the renewable energy. Its main goal is to develop a conceptual and infrastructural approach to allow efficient management of higher amounts of renewable energy and balancing of supply and demand in electricity networks. MIRACLE acknowledges the fact that very often the production from RES cannot be controlled and planned. Instead, some part of the demand can be shifted to times when the production from RES is available thus facilitating more efficient use of intermittent RES generation. The project leverages this approach based on an active consumer involvement in energy markets and the demand-response principle [THL10]. Here, a consumer can take an action in response to particular conditions within the electricity system (such as overproduction or underproduction from RES).

The MIRACLE project is subdivided into eight different work packages, where each of them is devoted to a particular research or management focus. This report is an integral part of work package 3 (WP3), which focuses on challenges of data collection and analysis encountered within the scope of MIRACLE. The report is a product of the initial task T3.1 of the WP3. It presents the relevant state-of-the-art that addresses the data management issues of MIRACLE. This report will serve as an input for the following tasks of the WP3, where the presented state-of-the-art will be further advanced to tackle MIRACLE-specific research issues.

Section 3 of this report briefly introduces the MIRACLE's scenario and presents the requirements for the data management system of MIRACLE that needs to be designed during the project. The following sections present the existing state-of-the-art that can be used to achieve these requirements. Specifically, Section 4 surveys the main categories of existing systems with regard to their applicability in future electricity data management systems, including the MIRACLE system. Section 5 presents the data analysis and query processing state-of-the-art that can handle the MIRACLE data, characterized by being uncertain, continuous, forecasted, multi-dimensional, and streaming. Then, relevant optimization techniques are presented in Section 6. Finally, Section 7 introduces existing computer systems in the energy domain that solves the MIRACLE-related electricity supply and demand controlling and planning issues. We conclude the survey by pointing out relevant research areas and further directions of investigations in the MIRACLE project.

3 Miracle Scenario

3.1 Overview

The electrical power available from weather-dependent RESs in many cases varies in ways that do not match the variations of the electricity demand. The weather cannot be planned, and thus weather-dependent electrical power often has to be given away due to a lack of demand or, in addition to the production from RESs, an energy from different (often not environment-friendly) types of sources has to be used to balance the demand.

However, some part of the demand (and the production from non-RESs), unlike the production from RESs, can be planned. For example, a consumer will always use the light when it is needed, but he may charge his electric vehicle at any time during the night as long as it is fully charged the next morning. In the former example, the consumer uses energy in an arbitrary manner; thus, this type of demand cannot be planned. However, the charging of an electric vehicle, for instance, can be scheduled to occur at a moment when surplus production from RESs is available. Figure 1 illustrates how demand flexibilities can be utilized to increase the use of electricity from RESs at a large scale. Here, dashed lines, solid-filled areas, and shaded areas depict the available production from RESs, non-flexible (unschedulable base-load), and flexible demand, respectively. The schedulable demand of one or billions of consumers can be shifted in time so that renewable power is used more efficiently (the top versus the bottom of the figure).

MIRACLE (Micro-Request-Based Aggregation, Forecasting and Scheduling of Energy Demand, Supply and Distribution) is an ongoing EU FP7 project that addresses the challenge of balancing electricity consumption and production by leveraging flexibilities in energy demand and supply. In MIRACLE, individual prosumers, i.e., entities that can both consume and produce energy, are allowed

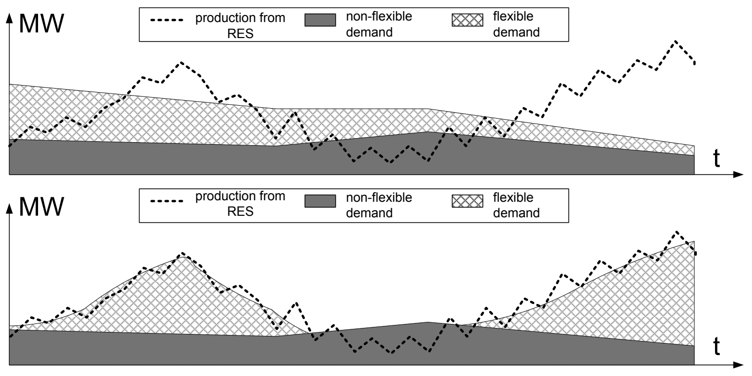

Figure 1: Balancing the consumption and the production from RES

to request and offer electricity with defined flexibility (e.g., flexibility across time). Such requests and offers are continuously collected and scheduled so that:

- Electricity demand in an electricity network better matches the supply.
- Production of electricity from large amounts of RESs is efficiently utilized.

In MIRACLE, advantages for both electricity producers and consumers, as well as for the environment, are envisioned:

- Producers can sell electricity for better price because the demand is more flexible (currently, wind-generated electricity must some times be given away for free because there is no demand).
- Consumers can get cheaper electricity if they are flexible regarding the time of consumption. The rescheduling possibility in MIRACLE allows reacting to problems in an electricity network better, thus reducing the amount of expensive control energy needed to balance the network in such circumstances. The control energy reduction together with the more intelligent use of renewable energy allow cheaper electricity price to be offered for electricity consumers (only to those that tolerate demand flexibilities).
- Much more renewable energy can be used in the electricity consumption, since much of the demand can be shifted to a time when, e.g., wind energy is available.

The central component in MIRACLE is a reliable, distributed, and highly scalable computer system infrastructure, which handles very large volumes of electricity-related data and provides services to very large numbers of users in near real-time. Its potential users are prosumers, electricity traders, and electricity network operators. Utilizing specialized hardware or software interfaces they all access the MIRACLE infrastructure through the public Internet, issuing queries and getting results from the infrastructure.

The following sections cover the conceptual architecture of the MIRACLE infrastructure and provides its potential use-cases.

3.2 Conceptual architecture

The MIRACLE infrastructure has a hierarchical architecture that mirrors the structure of the European energy market. The architecture contains sets of inter-connected electricity data management

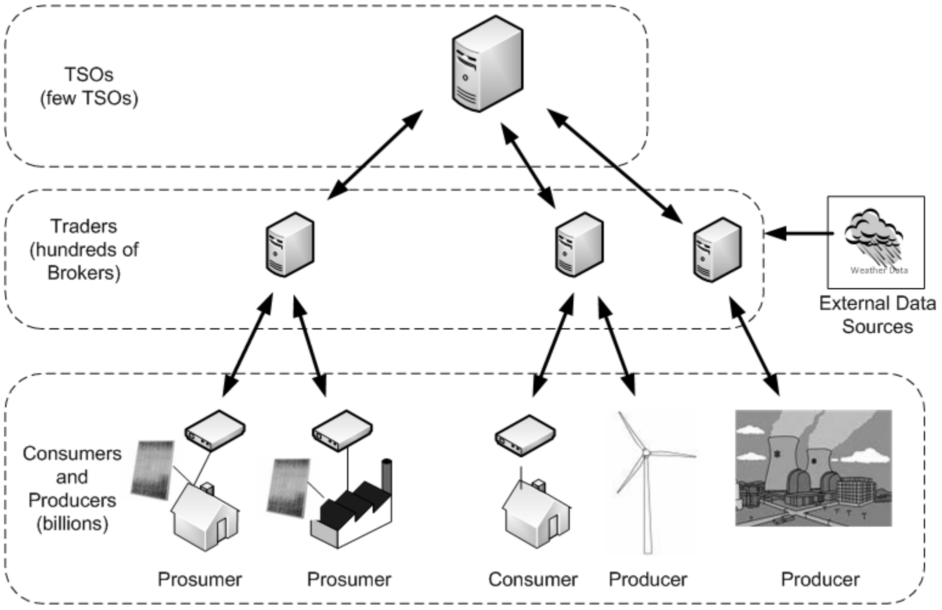

Figure 2: Miracle Architecture

subsystems (EDMSs) at different levels. These subsystems are used by different actors in the European energy system. Specifically, individual prosumers, traders, and transmission systems operators (TSO) use MIRACLE EDMSs at the first, second, and third level, respectively. Figure 2 visualizes the subsystems, hierarchical levels, and data flows (black arrows) within the MIRACLE infrastructure. Note, the infrastructure may span above the level of TSOs and can have more than 3 hierarchical levels.

EDMSs at the higher (traders, TSO) hierarchical levels control the consumption and production of lower level parties (prosumers, traders) through their respective EDMSs. The EDMS of an electricity trader or network operator functions similarly to a large-scale distributed control system with a feedback-loop (see Figure 3). It continuously and concurrently conduct the following tasks:

- In near real-time, it forecasts future power consumption and production in the electricity network (or sub-network) based on: (i) historical power measurements and computed forecasts (forecast of non-shiftable demand or supply) of the lower level parties (prosumers, traders) ; (ii) external data (e.g. weather forecast); (iii) scheduled and unscheduled requests and offers from the lower level parties.
- It collects consumption requests and production offers from EDMSs of the lower level parties.
- It schedules collected requests and offers so that the total consumption and production in the electricity network (or sub-network) follow a certain reference profile, e.g., zero profile for a balance in the system or a non-zero profile when there are imports/exports in the network (sub-network).

In MIRACLE, actuation of the energy balance in the electricity network is based on schedules that are delivered as the response to every demand request or production offer of the lower level parties. The feedback from the electricity network is provided as near real-time measurements and local forecasts of consumption and production.

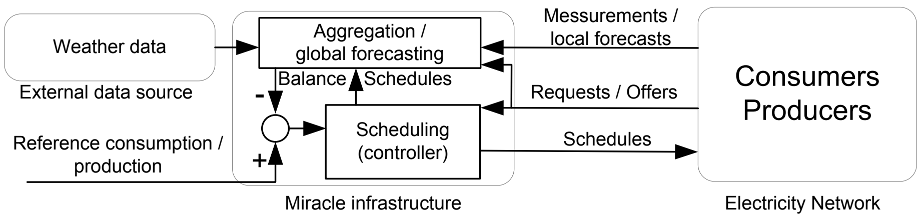

Figure 3: MIRACLE as a control system

A EDMS also incorporates external data such as wind speed, wind direction, and amount of sunlight. Such external data supports the prediction of future consumption and production and allow a party to forecast the demand and supply better.

3.3 Use cases of MIRACLE

The following subsections briefly cover MIRACLE use cases of prosumer, trader and system operator and provides an example scenario of the MIRACLE use.

3.3.1 Potential use of MIRACLE

Prosumer's use of MIRACLE

Every prosumer such as household, factory or power plant uses specialized local electricity data management systems (local EDMS). The local EDMS is assumed to be a cheap (relatively) physical device that connects to the MIRACLE infrastructure through the public Internet. It implements the intelligent power metering [MG97] function that allow monitoring consumption and production of individual appliances and store their energy consumption and production profiles. Also, the local EDMS is able to collect energy requests (requests for consumption or production) issued for their use and automatically turn individual appliances on or off at certain times. As an optional feature, local EDMS may also integrate some sort of consumption and production forecasting functionality.

Historical and predicted (if supported) electricity consumption and production data is continuously published to a EDMS at the trader level. The data specifies prosumer's actual or predicted power consumption and production, and MIRACLE primarily uses it to update the forecasts of total future demand and supply at the scale of the trader.

Energy request specifies prosumer's intention to use certain appliance (that generates or consumes electricity) within some flexible time interval. Such requests are typically collected automatically from individual appliances or specified manually on local EDMS by a prosumer. The local EDMS combines such energy requests with a power profile of the appliance, then generates and issues a special query to the MIRACLE subsystem of the trader level. The query is termed a *micro-request*, and it can specify: (i) a profile of electricity that is needed by a household in some defined time interval (ii) a profile of electricity that can be generated by household's power source in some defined time interval. They may also specify auxiliary data such as power generation price or information on if the power consumption or production can be interrupted and resumed afterwards.

Figure 4 depicts an example of data that is sent from a local EDMS to a trader level EDMS. Here, dark and light solid-filled areas visualize prosumers's historical and predicted power consumption, respectively. The shaded areas depict an energy consumption profile and its flexibility in time, captured by some micro-request. Here, the micro-request is issued for a use of an appliance (e.g.,

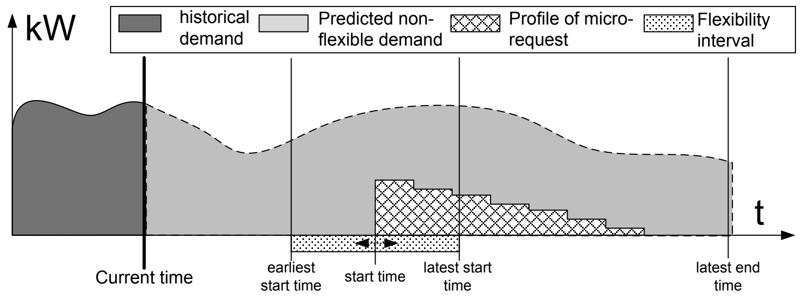

Figure 4: Example of the micro-request, historical and predicted electricity consumption

charging of electric vehicle) during the period from *earliest start time* to *latest end time*. With regards to given energy profile the appliance must start between *earliest start time* and *latest start time* to finish before the *latest end time*.

The trader level subsystem sends a reply to a micro-request. This reply is termed a *micro-assignment*. Micro-assignment is a schedule that indicates to a local EDMS when exactly a certain appliance has to start and eventually uses electricity (*start time* in Figure 4).

Local EDMS of a prosumer may also provide a user interface that allows to monitor the consumption and production, to insert energy requests, to view forecasts and the scheduled time for requests (micro-assignments).

Trader's use of MIRACLE

Traders ("balance responsible parties") use the MIRACLE infrastructure to plan and balance electricity at their level of the electricity network.

When trader's EDMS receives micro-requests for consumption and production from prosumers, it typically aggregates them into so-called *macro-requests*. These macro-requests are sent to electricity system operator's EDMS, where they are scheduled. For every macro-request, a so-called *macro-assignment* is generated and sent back to EDMS of the trader. Then, the trader's EDMS disaggregates these macro-assignments into a set of micro-assignments, which consequently are distributed to EDMSs of respective prosumers. This allows traders to shift demand and supply flexibly offering a value-added service for the electricity system operator. Traders can employ different request-aggregation and assignment-disaggregation schemas based on their individual business goals.

Once a trader buys or sells certain amount of electricity for certain time periods from another trader, it is responsible to consuming or delivering the contracted amount of electricity. Any imbalances of contracted and actually consumed or produced energy leads to financial penalties for the trader. Thus the trader can use MIRACLE to schedule consumption or production of end prosumers locally so that contracted amounts are met.

The MIRACLE infrastructure can be used to support trader's decision to generate energy locally or buy/sell energy from/to other traders externally. The infrastructure can provide information such as aggregated historical and forecasted power consumption and production of various prosumers, based on specified time interval, region, prosumer type, product type (e.g., RES).

System operator use of MIRACLE

Transmission system operator can utilize the MIRACLE infrastructure to stabilize the electricity network by requesting traders to shift some demand or supply through macro-requests and macro-assignments. For example, when traders use much power from RESs and deviate from their schedules due to the lack of wind, other traders may reduce a demand and help to maintain the balance in the grid through the rescheduling of micro-requests. This option is much cheaper comparing to the control energy (spinning power) which must be used for balancing demand and supply today. Hence, rescheduling will allow the operator to use more green energy and reduce amount of control energy needed to balance the network.

3.3.2 Example of the MIRACLE use

This section presents an use case example and possible interactions between the MIRACLE subsystems.

Assume a user owns an electric vehicle with a rechargeable battery. The battery requires $50kWh$ of energy and 2 hours of time to be fully charged. The charging power profile of the battery is monotonically decreasing and its reaches zero at hour 2. The user arrives home at $10pm$ and he wants to charge his vehicle's battery at the lowest possible price before the $7am$ of the next morning. Then, there is a sequence of actions that may occur in the MIRACLE infrastructure (see Figure 5):

1. The user attach the vehicle to a special (a local EDMS controlled) outlet. The system recognizes the vehicle, its power consumption profile, and assumes the charging completion time ($7am$) that may be preprogrammed in advance.

2. The local EDMS generates a micro-request based on the consumption profile at the charging time flexibility (shaded areas in Figure 6). The micro-request is then forwarded to the trader's EDMS.

3. The trader's EDMS schedules the request and replies a micro-assignment to the user's local EDMS for an electricity consumption at 1am. It is assumed that at $1am$ there is a surplus production from RES and the cheapest price can be provided.

4. The trader's EDMS updates macro-requests at the TSO's EDMS assuming that macro-requests deviated from the old ones substantially.

5. Assume that at $12pm$ the TSO notice upcoming shortfall of supply at $1am$-$2am$, e.g., due at the wind being less strong than expected, thus it issues a macro-assignment for the trader offering to reduce its demand with a specific amount at $1am$-$2am$ for a certain price.

6. The trader accepts the macro-assignment and reschedules respective micro-requests (if possible) from $1am$-$2am$ period to a new time. This results to a electricity price increase for the flexible producers and price reduction for the flexible consumers. The user's request was shifted from $1am$ to $3am$ providing a price reduction of €10/MWh.

7. A new micro-assignment is sent from trader's EDMS to the user's local EDMS.

8. The user's local EDMS starts supplying power for the electric vehicle at $3am$.

9. The battery of the electric vehicle is fully charged at $5am$.

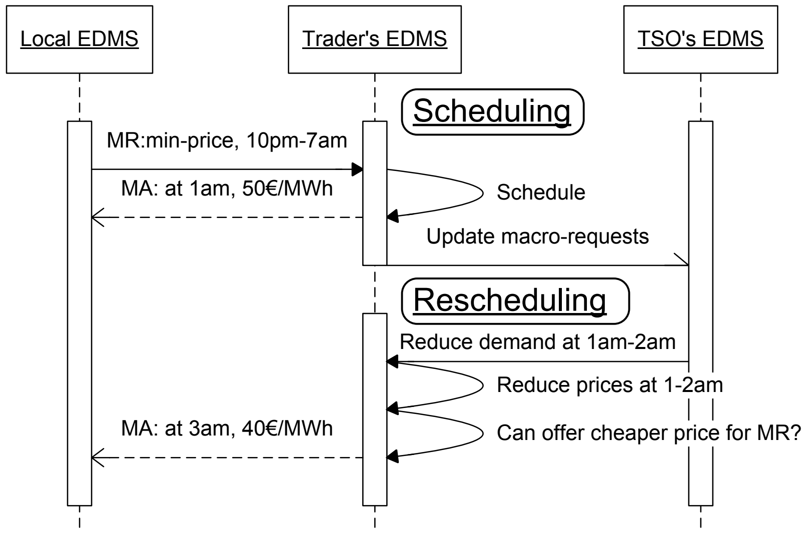

Figure 5: A sequence of actions in the MIRACLE example scenario

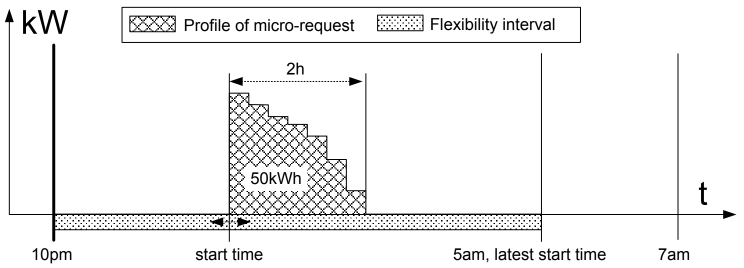

Figure 6: An example of the micro-request issued for the vehicle's battery charging

3.4 Uncertainties in MIRACLE

Different types of uncertainties of data are encountered in different domains. In MIRACLE we consider external data management domain and internal data management domain. Two primary types of data uncertainties, *inherent uncertainty* and *introduced uncertainty*, are identified in these domains respectively.

- *Inherent uncertainty*

  Inherent uncertainly emerges with a data that the MIRACLE infrastructure collects and integrates. Examples can be predictions of weather and electricity supply/demand. They have some level of uncertainty that is inherent by the nature of data. For example, as visualized in Figure 7, the factual electricity demand or supply can be measured accurately, but their future values can be predicted only with some probability (shaded area represent 95% confidence interval).

- *Introduced uncertainty*

  Often it is not feasible to track or compute certain values very accurately due to the communication or computation costs. In many cases, tolerating a certain level of uncertainty allows cutting down communication or communication costs by an order of magnitude. For example, an accurate computation of current-time electricity consumption in the electricity network considering billions of consumers would require an extremely high amount of communication and computation. However, computing the electricity consumption up to some accuracy would be substantially less expensive. Moreover, considering inherently uncertain data, such introduced uncertainty does not distort results of data analysis significantly if the introduced uncertainty is lower than the inherent uncertainty. Trading an amount of communication and computation for accuracy is an attractive option of the MIRACLE infrastructure design.

Evaluation of the uncertainties is a very important aspect of MIRACLE data analysis. For example, the MIRACLE infrastructure could provide a whole spectrum of values with their probabilities, i.e. probability distribution, as a reply to a query on expected demand, supply, or wind-speed at a certain time moment. The result is much more informative than a single average value. The notion of uncertainty in query results can, for example, improve a quality of trader's or system operator's planning decisions. In some circumstances the amount of uncertainty within a query result can be traded for query processing performance in cases when a rapid result for the query manipulating large amount of data is needed. The MIRACLE can provide an approximate result for a query and an estimate of an error in it. Consequently the MIRACLE infrastructure can support probability queries, when a probability for certain predicate to be *true* is needed, and handle queries with a user specified requirements for uncertainty.

Moreover, MIRACLE uses several types of data (both certain and uncertain) which varies over the time, e.g. forecasts of weather or electricity productions. Thus we have several versions of data representing the same fact, for example predictions of wind speed at a given hour were made 2 days ago, then updated 1 day ago, and then refined 2 hours ago. As a result, we have 3 database entries representing the wind speed at the given hour, but only the last one of them is valid. However, we must keep a record of all of them since forecasts of electricity supply were based on them. Consequentially, another type of complexity arises when we are dealing with multiple versions of the same fact.

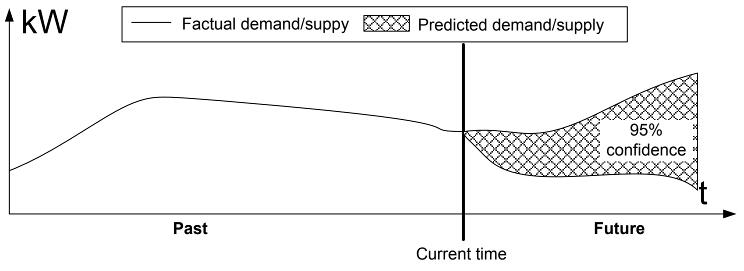

Figure 7: An example of factual and predicted demand and supply

3.5 Requirements for the MIRACLE infrastructure

This section covers functional and non-functional requirements for the MIRACLE infrastructure that will be developed during the MIRACLE project. These requirements have to be satisfied to be able to achieve the objectives and visions of the MIRACLE project resulting in successful management of electricity network domain's data and benefits for all stakeholders. The requirements for the MIRACLE infrastructure are presented in the list below.

Scalability requirements

- Scale up to the to level of continent (e.g., Europe) and handle requests from every household in it. It must be efficient enough to handle up to few billion connections from individual prosumers simultaneously and process up to few hundreds micro-requests from households per day (approx. 10^{11} micro-requests per day).
- Store historical electricity consumption and production data of various granularities (from various subsystems of Miracle) for certain time period, e.g. 1 year, resulting in approx. 100MB of data from consumer and 200PB of data in total.

Performance requirements

- Provide a time guaranties for a consumers. Time from issuing the micro-request until the micro-assignment is received should not exceed 2 minutes.
- Enable the trader and network operator to control electricity consumption and production in near real-time.

Data integration requirements

- Integrate various types of heterogeneous external data, e.g., speed and directions of wind, air temperature, to support a more accurate electricity demand and supply prediction.
- Support of discrete and continuous data flows. These may result from discrete micro-requests initiated by consumers, and continuous measurements of production and consumption in the electricity network.
- Store and manage multiple versions of the data such as forecasts.

- Support an uncertain data integration and handling. The uncertainty must be managed within data aggregation and query processing.

Massive Distribution Requirements

- Build a system with autonomous self-containing nodes, which are able to operate while disconnected from the other nodes.
- Maximize data consistency within these massively distributed systems. The main challenges lie in data propagation delays and temporarily off-line nodes.

The following sections cover existing state-of-the-art techniques and concepts that can be used to achieve the requirements above.

4 State-of-the-art in Data Collection and Integration

According to the introduced MIRACLE scenario, the conceptual architecture inherently requires data collection and integration from a huge number of consumers and producers in order to provide consolidated (global) views over the hierarchically structured subsystems.

Traditionally, we distinguish between horizontal integration and vertical integration within an information system pyramid. Horizontal integration refers to the immediate data synchronization between different operational systems of the same hierarchy level, while vertical integration refers to the data consolidation from operational into dispositive and strategic (analytical) systems from lower to higher hierarchy levels and thus, with a longer time horizon. The MIRACLE scenario can be seen as a use case for the current trend of operational BI (business intelligence) [DCSW09, O'C08, WK10], which requires technologies from both types of integration, in order to achieve high up-to-dateness of analytical query results over many source systems. These up-to-date query results are required for operational processing (e.g., forecasting and scheduling). As a result, a global view over the source systems with high data freshness requirements is needed within the MIRACLE scenario.

Essentially, there are two main approaches how to achieve such a global view over data of many source systems. According to the survey of Domenig and Dittrich [DD99], we distinguish between (1) virtually (logical) integrated systems, where global queries are rewritten to local queries to the source systems, and (2) materialized (physical) integrated systems, where the data of the source systems is physically consolidated into an integrated, redundantly stored database.

In this section, we give an overview over the state-of-the-art of data collection and analysis in the sense of query processing within both system categories. Furthermore, we review existing techniques from the areas of data exchange (data collection) and web scale data management systems. Note that the D1.1 State-of-the-art report and initial version of the architecture, roles and process model describes the overall MIRACLE system architecture with a focus on its structure. Obviously, this system structure has direct influence on query processing and thus the contents of this section as well. Finally, we will review the requirements and draw conclusions, which system type would be most appropriate for a request-based electricity data management system with regard to data collection and integration.

4.1 Virtually Integrated Systems

Virtual integration of systems refers to a form of logical integration, where the data resides within the source systems [ÖV99]. Then, global queries are rewritten to several local queries, which results are combined by a central mediator. Kossmann gave a survey [Kos00] of distributed query processing. Except for distributed caching, data is not materialized at the central mediator. This has several implications for the use of such systems. As advantages, such a system provides high flexibility and high up-to-dateness of query results because changes of the sources are directly included into the query results. However, there are disadvantages with regard to the query performance due to the ad-hoc integration of several sources. Furthermore, in order to ensure strong consistency, the two-phase commit (2PC) protocol (or the optimized variants 1PC and 3PC) is required and there is no data history.

Virtual or distributed database management systems are further separated into two main types. First, there are Virtual DBMS (VDBMS) that are designed in a top-down manner and the single source systems are homogeneous. The main reasons for such an architecture are availability (e.g., high availability and disaster recovery) as well as performance and scalability (e.g., local-based access). Second, Federated DBMS (FDBMS) are built in a bottom-up manner in order to integrate existing heterogeneous systems. Thus, the main reason for a federated architecture is the consolidation and interoperability of existing heterogeneous systems. In the following, we give an overview of current research directions for both areas.

There is plenty of related work on virtually integrated, homogeneous systems in the form of distributed database as well as parallel database. Please, refer to the survey of Özsu and Valduriez [ÖV96, ÖV97] for a detailed distinction between these two system types. Essentially, a distributed database provides a logical (virtual global view) over many homogeneous but distributed source databases with the aim of availability, location-based access and performance. In contrast, parallel DBMS aim to exploit parallelism in data management. For example, there are interesting approaches on load balancing in parallel DBMS [BFV96, DG92, RM95]. In conclusion, from the viewpoint of data collection and integration distributed databases are more relevant than parallel DBMS but their concepts can be extended with optimization approaches from the area of parallel DBMS as well.

In contrast to distributed databases that are based on a top-down design methodology, federated databases are used in order to integrate heterogeneous distributed systems including structured, semi-structured and unstructured data sources. Typically, heterogeneous systems are integrated with so called foreign data wrappers, which concept is also standardized within the part MED (Management of External Data) from the SQL standard [ANS05]. There, arbitrary external applications and systems can be integrated within global query processing. Examples for this system category are the research prototypes Garlic [JSHL02] as well as the products IBM federation server or Sybase ASE DTM (former known as component integration services).

For both types of virtually integrated and thus, distributed systems (which aim is to provide a transparent global view over multiple source systems), we observe important conflicts of goals. This conflict between the major goals of Consistency, Availability, and Partition tolerance has been explicitly formulated by Brewer within the CAP theorem [Bre00]. There, consistency refers to the atomicity property that all clients see the same data, availability means that all clients can access an available client, and partition tolerance represents the requirements of tolerance to temporary unreachable subnets. Finally, the theorem states that you can have at most two of these properties for any shared-data system. This claim was subsequently proven by contradiction [GL02].

After having reviewed the main categories of virtually integrated systems, we can conclude that for two reasons, these virtual integrated systems might not scale in the MIRACLE scenario with millions of source systems. First, virtual data integration requires consistency in the sense of transaction atomicity. Due to the CAP Theorem, this might lead to blocking subsystems (unavailability) or partition intolerance (partitioned subnets cannot handle the task of scheduling). Second, the virtual integration of this huge number of source systems by itself might lead to low performance and scalability.

4.2 Materialized Integrated Systems

In contrast to virtually integrated systems, within a materialized integrated system, the data of the source systems is consolidated, cleaned and redundantly stored within read-optimized data management systems and global queries are only executed on this consolidated database.

4.2.1 Overview

This concept of materialized integrated systems is typical for data warehouse (DWH) infrastructures, which are especially advantageous in terms of query performance because specific read-optimizations can be applied. In addition, the performance of source systems does not suffer from global queries and we can establish additional functionalities such as data history, archiving and consolidation. The drawbacks are the required synchronization between the source systems and the data warehouse and thus, typically the up-to-dateness is not as high as for virtual integrated systems. Furthermore, also the redundant storage of data imposes overhead.

As already mentioned, data warehouse infrastructures are a representing example of materialized integrated systems. In this OLAP (online analytical processing) context, both row-oriented DBMS as well as column-oriented DBMS are applied. Note that both have advantages and disadvantages [HLAM06, AMH08] and thus, the decision should be made with regard to the predominant workload. However, based on the argument that each application context might require its own tailor-made data management system [SMA$^+$07], recent research especially, focused on column stores in order to achieve high performance and high compression rates for analytical workloads.

4.2.2 Column Stores

Essentially, a column store uses vertical fragmentation of relations in order to speed-up analytical queries. As an advantage of this vertical fragmentation, better compression rates are achievable because all data of a page belong to the same attribute and hence, the same domain. Furthermore, fewer data must be read when accessing only a subset of columns. In contrast, conceptually, additional join operations are necessary in order to reconstruct the individual tuples and the update performance suffers from this read-optimized data layout. In conclusion, column stores are especially, advantageous for analytical queries, where huge amounts of data from only few attributes are read, aggregated and only few tuples are returned as a result.

Example systems for this category of column-oriented DBMS are the commercial products Sybase IQ [MF04], SAP BIA [LLR06], and Vertica. In addition, many research results have been published concerning research prototypes such as MonetDB [MBK00, BZN05] and C-Store [SAB$^+$05]. Current research mainly focus on specific compressions techniques [AMF06, WPB$^+$09, BHF09, Aba07], query processing and optimization [BZN05, IKNG09], tuple reconstruction [AMDM07, IKM09], as well as adaptive storage re-organization [KM05, IKM07a, IKM07b, IKM09].

In conclusion, column stores are best suited for analytical workloads with long running aggregation queries and only few updates. However, the MIRACLE scenario exhibits many updates in terms of a stream of incoming observations and requests. This might not be problematic if these updates are append only and affect only time series data. In this case, column stores are well suited for our scenario. Otherwise, the high update rates would reason the use of traditional row-oriented DBMS. An alternative solution is the use of the ephemeral data maintenance techniques [SJ06, SJS06]. They are based on data tagging with expiration times that indicate when tuples cease to be valid in the database. The expired tuples are kept invisible to the user and are removed physically in a delayed fashion. This allows removing tuples in bulks which can boost the overall data updating performance in column stores. In any case, due to the maintenance of long data histories, time series

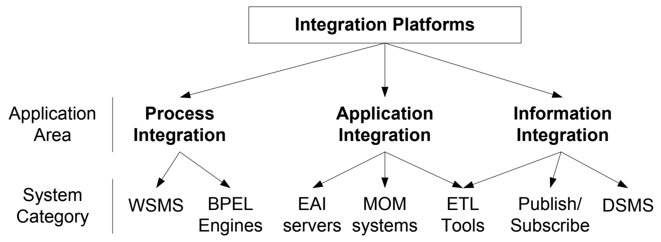

Figure 8: Integration Platforms for Materialized Integration

and the integration of a huge number of source systems, a materialized integrated system is most appropriate for MIRACLE.

4.3 Data Exchange

In case of materialized integrated systems—that appear to be most appropriate—as described in Subsection 4.2, efficient data synchronization between the huge number of source systems and the consolidated data warehouse is one of the major challenges. Thus, in this subsection, we will survey the major categories of systems for data exchange (collection and synchronization).

From an integration point of view, we distinguish the four main application areas of (1) information integration, (2) application integration, (3) process integration and (4) GUI integration. In this context, Figure 8 illustrates the main system categories that are relevant for the MIRACLE scenario. There, we omitted the category of GUI integration (e.g., Portals and Mashups), as well as peer-data management systems and distributed database systems (VDBMS/FDBMS) because they use virtual integration approaches, while we focus on integration platforms for materialized integration only. First, there are systems from the area of information integration, which typically, use query-based integration of data-intensive systems. Examples are DSMS (data stream management systems), publish/subscribe systems as well as ETL tools (extraction transformation loading). There, data changes of the source systems are immediately propagated to the integration platform and forwarded to the target systems using continuous queries or subscription trees. Second, there is the category of application integration, where heterogeneous systems and applications are integrated by data-driven or time-based integration flows. Examples are ETL tools, MOM systems (message-oriented middleware), and EAI servers (enterprise application integration), which have converged more and more in the past [HAB$^+$05, Sto02]. Third, the area of process integration refers to the integration of homogeneous services (e.g., Web services) with integration flows.

We will concentrate exclusively on the three main types of EAI servers, ETL tools and DSMS because these subsume MOM and publish/subscribe systems as well. Due to its limited applicability and performance issue, we do not consider categories from the area of process integration. Further, also publish subscribe systems are not considered, because in the MIRACLE scenario only a single subscriber would be present at each organizational hierarchy level. In the following, we survey the categories of ETL, EAI and DSMS in more detail.

4.3.1 ETL Systems

ETL (Extraction Transformation Loading) is a wide research area with many different facets. As a result, many different ETL tools and approaches exist. However, typically ETL integration flows are used in order to specify and execute data-intensive integration tasks, where data from many source systems is extracted, transformed and finally loaded into the data warehouse infrastructure. Essentially, such an ETL flow is a data flow graph of operators. Dayal et al. gave an overview on characteristics of ETL flows [DCSW09, SWCD09].

As one of the most important research directions of ETL systems, there is a trend towards operational BI (Business Intelligence), where data changes of the operational source systems are directly propagated to the data warehouse in order to achieve high up-to-dateness for analytical query results [DCSW09, O'C08, WK10]. This requirement is typically addressed with one of the following two strategies for real-time ETL [DCSW09]: First, the frequency of periodical delta load is simply increased (near real-time). Second, data-driven ETL flows are used, where data changes of the operational source systems are directly propagated to the data warehouse (so-called trickle-feeds). As a result of both strategies, many independent instances of ETL flows—with rather small amounts of data per instance—are executed over time. For these reasons, there are high performance demands on the ETL systems.

This is similar to the real-time propagation of requests within the MIRACLE scenario in order to enable active customer involvement and the integration of more renewable energy sources. Therefore, the research area of real-time ETL is relevant for future electricity data management systems as well. For this reason, in the following, we will have a closer look at real-time ETL approaches.

Recent research results within the domain of real-time ETL can be classified into the following three categories:

- *ETL Flow Optimization:* Similar to query optimization in DBMS, the ETL flow optimization mainly focus on rewriting logical and physical plans in order to optimize it for a certain optimization objective such as performance.
- *Incremental Maintenance:* The category of incremental maintenance focus on efficient incremental update of the DWH and the maintenance of materialized views in the presence of high update loads.
- *Query Scheduling:* Further, query scheduling refers to the priorization of queries and updates in order to achieve better performance, higher data freshness or fair query response times.

In the following, we review these three categories in more detail and put existing work into this classification.

With regard to ETL flow optimization, Simitsis et al. proposed the state-space optimization of ETL flows [SVS05a, SVS05b] and discussed how to decide on the physical implementation of ETL flows [TVS07]. While these approaches focused exclusively on the performance of ETL flows, the so-called QoX-driven ETL optimization [DCSW09, SWCD09] tries to provide an optimization framework for arbitrary optimization objectives such as performance, reliability, recoverability, or data freshness. However, so far no automatic optimization is supported by this framework except for selected aspects such as the optimization for recoverability [SWDC10].

While the ETL flow optimization addresses only the central ETL tool, incremental maintenance mainly focus on the incremental update of materialized views within the DWH in order to allow for high update performance in the presence of many updates. Essentially, we classify existing techniques into synchronous [AASY97, GM95] and asynchronous [SBCL00] approaches. We will focus on the details of materialized view maintenance when discussing optimization aspects in Section 6.

Most of recent research focus on different possibilities of query scheduling. The RiTE middleware [TPL08] provides up-to-date data on demand, while ensuring efficient inserts with bulk load approaches by deferring inserts until they are requested by any query. Furthermore, there are approaches that try to schedule queries according to a certain metric. For example, the stretch metric is used in order to provide fair response times according to estimated runtime of certain queries

[GMWD09]. Similar to the QoX metric suite, Thiele et al. modeled the scheduling of queries and updates as a multi-objective optimization problem [TFL09, TBL09] between quality of data (data freshness) and quality of service (query response times).

In conclusion, in order to allow for active customer involvement and real-time scheduling of energy demand and supply in future electricity data management systems, techniques from real-time ETL should be used as a conceptual foundation. For example, the application of query scheduling techniques might be advantageous due to different time horizons of requests such that we can benefit by priorization of requests. However, in contrast to real-time ETL, it is not enough to just propagate data in near real-time. Future electricity grids, have additional requirements. First, procedural aspects are needed due to rather complex processes between the different organizational roles of the energy market. Second, beside the real-time data exchange, also real-time data processing based on the propagated data is needed.

4.3.2 Enterprise Application Integration

The first additional requirement of procedural aspects of integration flows inherently leads to the category of EAI servers (Enterprise Application Integration). Similar to ETL tools, integration flows are modeled and executed by a central integration platform. The major difference is that EAI integration flows are typically control flow graphs of operators that allow for complex procedural modeling in addition to the pure extraction, transformation and loading of data.

The core concepts of EAI servers are (1) the combination of control-flow- and data-flow-oriented operators in order to allow for procedural integration flows and (2) the use of so-called inbound and outbound adapters that hide syntactic heterogeneities of external systems and applications, while all operators can work on a common internal message representation.

Current research mainly focus on the optimization of these integration flows, where we distinguished existing approaches into the following three main categories:

- *Data Transfer Optimization:* The access to external systems and applications is typically time-expensive and this, it is tackled with data transfer optimization, which mainly focus on streaming transferred data (unchanged amount of exchanged data) or reducing the amount of transferred data.
- *Static Operator Reordering:* The category of static operator reordering refers to the rewriting of procedural plans once during the initial deployment (optimize-once model).
- *Dynamic Operator Reordering:* In addition, to static operator reordering, dynamic reordering refers to the cost-based optimization of plans that enables the adaptation to changing workload characteristics.

According to data transfer optimization, two research directions are notable. A first group of approaches use a streaming invocation of Web services in order to increase the overall message throughput by increasing pipeline parallelism. For example, Srivastava et al. introduced the adaptive data chunking [SMWM06], where batches of tuples (chunks of messages) in the form of individual messages are sent to the external systems. This concept was refined by Gounaris et al. to the use of an online extremum-control approach [GYSD08b, GYSD08a]. In contrast, Preissler et al. introduced the concept of stream-based Web services [PVHL09] over multiple process instances without the overhead (network latency) of passing individual messages. While streaming does not affect the amount of exchanged data, a second group of approaches try to reduce transferred data. For example, BPEL-DT and BPEL/SQL achieve this by using references to data sets instead of physically exchanging data. BPEL-DT [HRP+07] reduces the transfered data by passing references to a data layer and using ETL tools when possible. Furthermore, Vrhovnik et al. introduced the rule-based optimization of BPEL/SQL processes [VSES08, VSS$^+$07], where several rewrite rules were defined in order to condense sequences of SQL statements and to pushdown certain operations to the external DBMS.

In addition, many approaches analyze dependencies between tasks of an integration flow and then rewrite tasks with no dependencies between them to parallel sub flows. For example, Li and Zhan determine the critical path of an workflow with regard to the execution time and then optimize only this part of the workflow [LZ05]. Typically, rewriting rules are defined in terms of algebraic equivalences [HML09, YB08]. In addition, the XPEDIA system [BABO$^+$09] achieves partitioned parallelism by partitioning large XML documents into multiple parts, evaluating the parts in parallel, and finally, merging the results. Furthermore, Srivastava et al. proposed an algorithm for finding the best plan of Web service calls (control-flow semantics) with regard to highest parallelism and thus, lowest total execution time [SMWM06].

Finally, there is the category of dynamic operator reordering, which is based on the observations of changing workload characteristics and on the problem that many optimization decisions can only be made in a cost-based manner based on execution statistics. In this context, the periodical re-optimization [BHW$^+$07, BHLW08] has been proposed as well as several specific optimization techniques such as flow vectorization [BHP$^+$09b, BHP$^+$09a], multi-flow optimization [BHL10], or message indexing [BWHL08].

In conclusion, the complex processes in future electricity grids can be expressed with procedural integration flows during data exchange using EAI servers. In this area, several optimization approaches have been proposed with regard to streaming and reducing the amount of exchanged data. Another approach for minimizing this amount of transfered data might be aggregation. However, in addition to the real-time data exchange, real-time data processing based on the exchanged data is required as well.

4.3.3 Data Stream Management Systems

In contrast to real-time data exchange with ETL tools or EAI servers, DSMS (Data Stream Management Systems) do not mainly focus on efficient data exchange, but on efficient processing of continuous queries (CQs) over streaming data. Therefore, this type of system allows to process data in real-time as data is propagated to this system. In future electricity grids, this is essential in order to integrate forecasting and scheduling into a real-time overall architecture.

Essentially, plenty of research prototypes and commercial products are available in the context of DSMS. Examples for this kind of systems are CAPE [ZRH04, RDS$^+$04, LZJ$^+$05], NiagaraCQ [CDTW00], StreaMon [BW04], PIPES [CKSV08, KS09, KS04], and QStream [SBL04, SLSL05]. All of these systems follow a similar approach, where continuous queries with time window semantics of operators read streaming data and typically aggregate or filter these streams efficiently. Thus, in contrast to traditional DBMS, data is transient and overload scenarios are simply addressed with load shedding techniques.

Two important research directions in this area of DSMS are (1) adaptive query processing (optimization during query runtime) and (2) load balancing and query partitioning over multiple server nodes.

DSMS typically use the following adaptation approach: The optimizer specifies which statistics to gather, requests them from the monitoring component, and re-optimization is triggered periodically or whenever significant changes of statistics have occurred [BB05]. Rewriting CQs requires (1) state migration (e.g., tuples in hash tables) [ZRH04] to prevent missing tuples or duplicates and to ensure the tuple order, or (2) flushing the pipelines of the continuous data flow graph. Hence, re-ordering of streaming operators is only applied in combination with extensive statistic profiling (e.g., for conditional operator selectivities) [BMM$^+$04]. As an alternative to this optimization model, routing-based adaptation can be applied, where no predefined plans but a central routing policy is used to process the streams of tuples. Examples for this adaptation model are eddies [AH00] and the self-tuning query mesh [NRB09].

Load balancing in DSMS is typically realized with *query plan partitioning* or data stream partitioning. Balazinska et al. used query plan partitioning [BBS04] in combination with the so-called box-splitting technique, which moves the data stream load across different nodes. Furthermore, Shah et al. presented a dynamically adjustable FLUX operator [SHCF03] that realizes execution-aware data stream partitioning. Data stream partitioning was also employed by Ivanova et al., where a window split strategy partitions the stream in order to combine results efficiently [IR05]. In addition, the query-aware plan partitioning [JMSS08a, JMSS08b], where operators of a continuous query are distributed over multiple nodes, was proposed with the aim of minimizing the transfered data between nodes.

In conclusion, especially from the perspective of real-time data processing, DSMS are also relevant for real-time electricity data management systems. However, additionally, persistent historic data, real-time data exchange functionalities and specialized query processing such as forecasting and scheduling are required. As a result, it seems to be advantageous to use concepts of data stream processing in combination with techniques from the areas of materialized integrated systems and data exchange in order to build a hybrid electricity data management system.

4.4 Web-Scale DBMS

In addition to the traditional integration approaches, we also review the state-of-the-art of web scale data management [Mel09], used within the area of cloud computing that has been described in the D1.1 State-of-the-art report in very detail. Both areas of request-based electricity data management and web scale data management, exhibit similar characteristics. First, in both areas, vast amounts of data from several sources must be processed. Second, this data is inherently partitioned according to the source systems. Third, due to the processing of the huge amount of data, computation is distributed across many server nodes, where the data partitioning enables simple distribution. In conclusion, concepts from the area of web scale data management might be applied in the context of electricity data management as well.

Typically, we distinguish web scale data management approaches into the two major categories of operational and analytical processing. Operational processing means that many operations per second, with only few rows per operations are executed. In contrast, analytical processing refers to a low number of operations per second but with billions of rows per operation. In the MIRACLE project both types of workload exist on different levels of the hierarchy. Therefore, in the following, we review both categories in detail.

From an operational perspective, there is an increasing need for efficient data management with fairly simple usage patterns. Due to the management of heterogeneous data and the simplicity of required queries, typically, so-called key/value stores or distributed hash tables (DHT) with simple put/get interfaces are used. In order to enable availability and scalability these systems are usually designed as distributed systems. Examples for this system category are Amazon S3 (Simple Storage System), Amazon Dynamo [DHJ$^+$07], Yahoo! PNuts [CRS$^+$08, SCS$^+$08] and Google BigTable [CDG$^+$06, CDG$^+$08]. Further, Brantner et al. demonstrated how to map database components to Amazon S3 [BFG$^+$08]. Essentially, there are many similarities with traditional concepts such as Peer-to-Peer (P2P) systems, or scalable distributed data structures.

Similar to that, also for analytical processing, scalability is the main reason for the distributed architecture. However, due to the analytical processing of mass data with a few long-running queries fundamental different approaches are used. Essentially, all analytical solutions use distributed data flow graphs in order to process data. Examples for this execution model are the frameworks MapReduce [DG04, Dea07], Hadoop, and Dryad [IBY+07, YIF+08]. In addition, there are extensions such as Sawzall [PDGQ05], PigLatin [ORS+08, GNC+09], and Scope [CJL+08, ZLC10] that built on top of the basic execution model but uses some kind of scripting language in order to reduce the development effort and allow for automatic optimization. Interestingly, Pavlo et al. argued that the MapReduce programming paradigm is similar to the execution model of parallel DBMS and they demonstrated that both system types have there advantages and disadvantages [PPR+09]. In consequence, HadoopDB [ABPA+09] aimed to create a hybrid that combines the advantages of both approaches.

For both operational and analytical processing, scalability is the key goal that reasons the distributed architecture. In conclusion, this system categories exhibit similar requirements as a electricity data management system. However, in addition to the processing of huge amounts of data, the MIRACLE scenario requires not only high throughput but also low latency in order to meet the real-time requirements.

4.5 Conclusions of Data Collection and Integration

To summarize, we surveyed the main categories of virtual and materialized integrated systems with regard to their applicability in terms of an architecture for data collection and analysis in future electricity data management systems. Furthermore, we reviewed recent research results from the areas of data exchange and web scale data management. Combining this classification with the described MIRACLE scenario, we can draw conclusions on the required system architecture.

First, the huge number of source systems in terms of energy consumers and producers, lead to a materialized integrated system. However, the existing organizational hierarchy reasons a hierarchy of systems. As a result the materialized integrated system is inherently distributed, which is similar to so-called staged or multi-tier data warehouse infrastructures. The main difference is that, within the MIRACLE scenario, the number of system nodes is several orders of magnitude higher than in existing multi-tier infrastructures, which poses a major challenge with regard to system scalability and consistency.

Second, a materialized integrated system requires data exchange between the systems of the hierarchy. Future electricity grids aim to involve the customer actively, which is similar to the trend of operational BI but with the difference of not being query-driven. For this reason, techniques of real-time ETL data propagation and priorization are required. Due to rather complex processes within the energy domain, procedural aspects of integration flows from the area of EAI can be adapted. However, in addition to active data propagation (ETL, EAI), data processing within an electricity data management system must be triggered actively as well. For this reason, additional concepts from the area of DSMS in terms of real-time analysis but with the requirement of persistent historic data is needed. Finally, current research on scalability of distributed systems that process huge amounts of data can be adapted from the area of web scale data management system.

In conclusion, there is plenty of existing work on efficient, real-time data collection and analysis. However, there is no one-size-fits-all system [Sc05]. In order to address the two major challenges of future electricity grids, namely (1) the active customer involvement and (2) the integration of more renewable energy sources, a new design for a tailor-made electricity data analysis and collection system is required in the form of a hybrid architecture of materialized integrated system and real-time data streaming management system.

5 State-of-the-art in Data Analysis and Query Processing

5.1 Handling of Uncertainties

Management of uncertainties is an essential aspect of MIRACLE data analysis and query processing (as described in Section 3.5), therefore in this section uncertainty management state-of-the-art techniques relevant for the MIRACLE scenario will be surveyed.

5.1.1 Overview

The management of data uncertainty has attracted considerable attention over the last few decades and is experiencing revived interest due to the number of new real-world applications, demanding support for managing, storing, and querying uncertain data [Agg09]. Examples include applications of information extraction, data integration, sensor network, mobile object tracking, and business intelligence. They manage data that is often associated with uncertainty because: (i) inherent ambiguity of the data (e.g., as ambiguity of the natural-language, forecasts); (ii) measurement inaccuracy; (iii) sampling discrepancy; (iv) outdated data sources; (v) various errors. In some applications, such as privacy, it is a requirement that the data has to be less precise and imprecision is purposely inserted to hide sensitive attributes of individuals so that the data may be published.

The research field of uncertainty handling mainly focuses on the modeling, management, and mining [AY09] of data with underlying uncertainties. It provides solutions to a number of problems such as uncertain data representation, collection, querying, indexing, and data mining. The existing works within this field can be classified based on different aspects such as (1) the domain and type of uncertain data, (2) types of uncertainties being addressed, and (3) methods used for capturing the uncertainty. Table 1 presents further classification of the existing work. The works may focus on: (i) uncertainties in discrete or continuous domains; (ii) uncertainties encountered when using various types of data such as relational or multi-dimensional; (iii) types of uncertainties such as inconsistency, imprecision, or ambiguity; (iv) fuzzy or probabilistic models used to capture an uncertainty.

The uncertain data in MIRACLE (e.g., supply, demand, wind speed, and temperature) is continuous, multi-dimensional, and streaming. For example, forecasts of electricity supply or demand can be linked to a specific location (e.g., electricity network or sub-network), type of producer or consumer, and time, and can be modeled as continuous variables that exhibit many updates in terms of streams of measurements. Another important property of the MIRACLE data is that it is often produced by aggregating other data, which is uncertain in many cases. Therefore, state-of-the-art probabilistic databases, OLAP, and data streams, all of them using continuous, multi-dimensional or aggregated data, will be reviewed in the following sections.

5.1.2 Probabilistic databases

A probabilistic database management system can store large volumes of probabilistic data and supports complex queries over it [DRS09]. From the perspective of uncertainty handling, the existing works on probabilistic databases deal with the representation and management of *tuple* or *attribute-level* uncertainties [SMS$^+$08]. Applications with categorical uncertainties use a tuple representation model, i.e. the presence of a tuple in a database is probabilistic. In contrast, attribute-level uncertainty models consider that a tuple is definitely part of the database, but one of more of its attributes is (are) not known with certainty (are imprecise or vague). Such attributes are represented either as continuous ranges or list of discrete alternate values. Generally, probabilistic databases, which can

Class	Variants
Domain	discrete, continuous.
Type of data	relational, multi-dimensional, XML, stream, spatial, spatio-temporal.
Type of uncertainty	tuple uncertainty, attribute level uncertainty; inconsistency, imprecision, vagueness, uncertainty, ambiguity (see [Mot97]).
Modeling of uncertainty	fuzzy, probabilistic.

Table 1: Existing work in the field of uncertainty handling

handle continuous domains, can also easily capture the case of discrete uncertainty. Similarly, probabilistic databases, which operates on discrete domains, are able to handle continuous uncertainty by, for example, sampling the continuous probability distribution. However, the latter one introduces a trade-off between accuracy (lots of samples) and efficiency (fewer samples). It also requires adding many tuples in a database in order to model a single uncertain value, which is not scalable. In MIRACLE, uncertainties in continuous rather than discrete parameter values are dominant and the continuous values of attributes rather than attribute existences are uncertain. Therefore, the work on probabilistic databases focusing on the attribute-level uncertainty in continuous domains will be covered.

Orion V2 [SMM+08] is a state-of-the-art uncertain database management system having a built-in support for uncertainty at the database level. It is relevant for the MIRACLE in the sense that it supports attribute-level uncertainty over continuous data types (the tuple uncertainty and discrete data types are also supported). From the perspective of the attribute uncertainty, Orion models it with probability density functions and offers special support for typical infinite distributions: Gaussian, Poisson, Binomial, or Bernoulli. When underlying uncertain data cannot be represented using one of the standard distributions, Orion allows capturing the specific distribution with histograms by enumerating values to represent the distribution. Orion uses efficient data access methods based on gathered statistics over probabilistic data, R-trees, signature trees, inverted indexes, and allow querying uncertain data efficiently. Probability density functions are used in Orion to reflect uncertainty in the results of queries that involve uncertain attributes. Although Orion can store attribute uncertainties over continuous data, the implementation only supports select-project-join queries and lacks support of the uncertain data aggregation (i.e. "group by" queries), which is crucial in the MIRACLE scenario. The probabilistic data management system *Trio* [AW09] handles uncertainty with a technique similar to *Orion V2* and supports select-project-join, but no aggregate queries.

Cheng et al. [CKP03] manage attribute-level uncertainty in the continuous domain by treating an attribute value as a continuous random variable. They characterize the uncertainty with an interval and a probability density function that bounds the uncertain value and specify its distribution inside the interval. The classes of probabilistic queries are identified based upon the nature of the query result and whether or not aggregation is involved in evaluation of the result. Then, they discuss query processing, query result quality measurement, and the quality improvement techniques for each of these different kinds of queries. In the context of MIRACLE, the particular class of value-based aggregate queries is relevant. This includes average, summation, minimum and maximum value probabilistic queries. A bounded probability density function is used to specify uncertainty within a result of such queries. The uncertainty model proposed in this paper is being used in other works on: (i) indexing of uncertain data [CXP+04]; (ii) processing of probabilistic nearest-neighbor queries with tolerance constraints [CCMC09]; (iii) processing of continuous probabilistic queries [ZCC10]

5.1.3 Uncertainties in OLAP

Systems in an OLAP setting use a multidimensional data model and offer high performance for aggregation queries. The typical multidimensional model in OLAP captures the data in which the *dimensions* are structured hierarchically and *facts* map to points in the corresponding multidimensional space. However, when the assumption that all facts map to points is relaxed to allow facts map to regions (that are less accurate than points), a particular new kind of attribute uncertainty, called *imprecision in dimension values*, arises. For example, a fact may specify that a certain amount of electricity was produced from RES, without specifying the concrete type of RES. In the MIRACLE scenario, multi-dimensional data (both certain and uncertain) and *aggregation queries* are dominant, thus the recent state-of-the-art on handling the dimension value imprecisions and attribute-level uncertainties in the continuous domains will be presented.

Burdick et al. [BDJ+05] manages both the dimension value imprecision and the attribute-level uncertainty (on values of measure attributes) in the OLAP scenario with a focus on aggregation query processing. First, three general criteria (*consistency*, *faithfulness*, and *correlation-preservation*), which must be satisfied by any approach handling uncertain data in an OLAP setting, are introduced. Then, two criteria satisfying approaches, *facts allocation* and *uncertain measure aggregation*, are proposed to handle the dimension value imprecision and the attribute-level uncertainty. In the facts allocation, imprecise data, which is assigned to higher levels of the dimension hierarchy, is partially assigned to lower level leaf nodes based on specified weights. For the uncertain measure aggregation, they suggest using discrete probability distributions aggregation techniques studied in the statistical literature under the name of *opinion pooling* [GZ86]. Their proposed facts allocation techniques may be relevant for the MIRACLE scenario in a case of multi-dimensional data aggregation, however their uncertain measure aggregation technique may not, since it focuses only on discrete domains.

Timko et al. [TDP06] generalize the notion of measures in OLAP data model with probability distributions and propose new types of query processing techniques that operate on these probability distributions. The query processing technique supports *aggregation* and *probability* queries. These queries can, for instance, be used to ask for whole probability distributions ("how many cars are in a given street?") or for summaries about the distributions ("what is the probability that the number of cars in the street exceeds 50?"). In their model, an uncertain (measure) value in a fact is captured by attaching the fact with the corresponding dimension value from the measure dimension and specifying a probability. Intuitively, this approach allows modeling discrete, but not continuous, uncertain measurement values. Comparing their approach to the probabilistic databases, the *attribute-level* uncertainty here is captured by uncertainty on a tuple that has additional attribute for a value of the measure dimension. The advantage of their OLAP model is the support of partial containments [TTT+05] within dimension hierarchies. This allows nodes of higher levels (e.g., years) of the dimension hierarchy (e.g., time hierarchy) partially contain the leaf nodes (e.g., weeks) of lower levels. In their model, the partial containment between two dimension nodes is specified by a percentage (degree) of how much of values of one nodes are included in another node. In terms of query processing, they propose pre-aggregation of measurements based on probability distribution. That allows approximating higher-level aggregates efficiently both in time and space.

The OLAP-based framework Sampling Cube [LHY+08] manages uncertainty in sampling data. The sampling data represents only a subset of an original population and therefore has underlying uncertainty. The framework assumes a normal distribution of the population values and captures an uncertainty by attaching a confidence interval to a query result thus indicating the reliability of the result. Also, a technique is proposed, which 'expands' a query when an area of an OLAP cube with too few samples is queried. This allows gathering more samples thus reducing the confidence interval, i.e., improving the quality of the answer. Moreover, a cube compression scheme which is based on the quality of sampling estimates is proposed. It allows to provide nearly the same answers as the full sampling cube with much smaller computation and storage requirements. The sampling is a way to approximate large amounts of data, therefore we believe that the sampling and the ideas of the Sampling Cube framework can be used to support some part of uncertain data management in MIRACLE.

5.1.4 Uncertainties in Data streams

Data stream management systems provide efficient, real-time processing of continuous queries over streaming data (as described in Section 4.3.3). In the area of uncertainty handling, some of the existing works focus on such systems that manage data streams with underlying uncertainty. Here the data is characterized by being inaccurate or misleading. First, such streaming data is pushed to a system, where it is instantly used for the query processing. Then, if no additional data storage facility is used, the data is typically discarded. Therefore, in case of the MIRACLE infrastructure, usage of a probabilistic data management system, must be combined with probabilistic databases or warehouses to be able to both process and store uncertain streaming data. The following paragraph present a typical state-of-the-art uncertain data stream management system.

The probabilistic data stream management system PODS [TPL$^+$10] supports relational processing of uncertain data streams modeled using continuous random variables. Here, the uncertainty of data is captured with probability distributions that are transformed by every relational operator of the system and propagated to query results. The system uses a flexible data model and efficient data processing algorithms. The data model is based on Gaussian Mixture distributions that allows capturing continuous attribute-level uncertainties and enable fast relational processing of data. Under this model, two efficient aggregation and join operators over uncertain data are supported. For the aggregation, one exact and two approximate techniques for computing *sum* and *avg* are used. The techniques are based on the Fourier transform, which allows reducing computation complexity of the aggregated distributions. These techniques can be combined based on the accuracy requirements preserving high performance in the stream processing. For the join operator, the *equi-join* and the join that pairs two inputs based on inequality comparison are supported. PODS supports an exact computation of a join result and offers methods for pruning tuples with low existence probabilities. PODS's uncertain data model and supported aggregation techniques, which were designed for the use in real-time, can be applied to support the uncertain data management in the MIRACLE scenario.

5.2 Multi-Version Data Support

As described in Section 3.5, it is often important to preserve multiple versions of time-varying data (e.g., forecasts). Therefore we cover the state-of-the-art of multi-version data management.

Typically, time-varying data for which multiple states are recorded is termed temporal data; and collection of time-referenced data is called temporal database [LÖ09]. Generally, a temporal database management system supports some type of temporal data model and implements a query language to access temporal data. Some of commercial products are Oracle [MTM08], utility applications for MS SQL Server, DB2, and Sybase.

Data values in a temporal database are associated with timestamps of one of the types: valid time, transaction time or bitemporal [BJ03]. Valid time refers to the time when the fact, denoted by the data, was true in the real world; transaction time refers to the time when the data was recorded as current in the database; and a bitemporal timestamp captures both valid time and transaction time. In short, valid time is usually provided by a user while transaction time is automatically inserted by the DBMS. In MIRACLE, some data can be tagged with an expiration time and a transaction time to capture the evolution of the data. Thus we are considering bitemporal databases in the following.

The timestamps used for capturing valid and transaction time may have different data types. Data types include time intervals and so-called temporal elements, which are finite unions of time intervals. The former have fixed length and are simple to accommodate, while the latter are closed under intersection, union, and difference, which can simplify querying. In the context of MIRACLE, either data type may be useful, although simple interval timestamps may be preferable.

An important requirement of the MIRACLE infrastructure is high performance, and each of the system's components must help to achieve that requirement. Kimball et al. has proposed dimensional data modeling [KR02] as a basis for analytical systems that must deliver high performance. Specifically, Kimball et al. propose to use relational database technology and to design database schemas as so-called star schemas that consist of a so-called fact table with many-to-one references to a number of so-called dimension tables. Bliujute et al. [BSSJ98] identify several drawbacks of using star schemas for managing temporal data (e.g., performance issues with slowly changing dimensions and with state-oriented data) and propose temporal star schemas as a better performing alternative to regular star schemas.

An other requirement of MIRACLE is transparency and conformance to standards. Therefore the most appropriate query language for temporal data retrieval is (possibly modified) SQL. Böhlen et al. [BJ03] analyze various approaches to offering support for temporal database management in SQL, covering temporal query languages such as SQL/TP, TSQL2, ATSQL, TempSQL. These languages represent different approaches that use different timestamp data types and that use different query language concepts for enabling the convenient querying of temporal data. Having in mind the nature of the temporal data in the MIRACLE scenario, usually the most recent data will be accessed and queries for selecting it will be performed. Several of the query language approaches and specific query languages offer good support for these needs.

In conclusion, the multi-version data integration and query processing in the MIRACLE scenario are very important aspects, and a solution that supports real-time data insertion and retrieval is mandatory. The existing proposals covered here are relevant in the MIRACLE context, but in order to use them they must be combined with uncertainty management methods.

5.3 Conclusions of Data Analysis and Query Processing

To summarize, we have analyzed several methods for uncertainty handling in heterogeneous MIRACLE data, and also reviewed some related work on multiple version data management. To conclude, the probabilistic data modeling based on probability density functions is typically used to capture uncertainty of continuous random variables, which can often be used to model data in the MIRACLE scenario. There are many solutions proposed which can be used in MIRACLE to implement probabilistic data modeling and support high-dimensional domains, data aggregation and multi-versioning, and data streaming. However, further research must be conducted in order to build highly scalable data management system for the MIRACLE infrastructure.

6 State-of-the-art in Query Optimization

In addition to the already discussed aspects of optimizing data collection and integration tasks, plenty of work exist in the area of traditional query optimization as well. This includes but is not limited to work on (i) efficient execution environments (e.g., column stores, and efficient index structures) (ii) semi-manual query optimization (e.g., offline design advisor, materialized views), and (iii) transparent query optimization (e.g., online physical design tuning or optimization techniques such as join enumeration or early aggregation).

With regard to the MIRACLE scenario, in this section, we concentrate only on selected aspects of this wide research area of query optimization. In particular, we review existing work of aspects that might be advantageous for the hierarchical architecture of a future electricity data management system.

6.1 Overview

With regard to electricity data management in MIRACLE, the three major technical challenges of aggregation, forecasting and scheduling exist. Due to the scalability requirements mentioned in Section 3, optimization with regard to minimizing the amount of transferred data is strongly needed. In this context, we concentrate on the following optimization aspects:

- *Efficient Tracking of Measurements:* Similar to tracking mobile objects, the amount of transferred data can be reduced by deferring messages until significant changes have occurred or until the expected behavior has changed. In this category, we review recent research results such as prediction-based tracking.
- *Early Aggregation:* Due to the hierarchy of distributed systems, the amount of transferred data can be reduced by early aggregation, where arbitrary aggregation dimensions such as time, customer, supplier, or product can be used. There, we focus on early aggregation strategies from a microscopic perspective (local query processing) as well as from a macroscopic perspective (in-network aggregation).
- *Materialized Views:* The forecast models and production/consumption schedules created upon the propagated data can be seen as materialized views with specific properties (e.g., uncertainty of predictions that allow for more aggressive optimizations w.r.t. model maintenance), where re-occurring queries use these views. Moreover, also the use of traditional materialized views might speed up query processing and reduce the amount of exchanged data. Thus, we review existing techniques of materialized view maintenance as well.

All these three categories of optimization aspects are relevant with regard to when and how to propagate data under the constraints of uncertain data. In the following, we present the state-of-the-art in these three optimization categories.

6.2 Efficient Tracking of Measured Values

In scenarios where a system continuously tracks the states of a large number of continuous processes the amount of incurred updates can be very high. To tackle the problem, efficient shared prediction based tracking techniques [CJNP04, JP07] were proposed. They allow reducing amount of state updates and thus the communication incurred by the tracking. The techniques assume a simple client-server architecture, where the server tracks the states of multiple processes with a certain minimum accuracy. Unlike time-based state tracking techniques where updates are sent to the server at regular time intervals, these are based on the prediction of the future state of a process. Here, a sensor device issues an update to the server and is aware of the server's prediction at any time. It issues an update only when the predicted state deviates by some threshold from the real state, obtained by the sensor. In the literature, this general concept of shared prediction based tracking is applied to the tracking of so-called moving objects such as vehicles. In this setting, *point-*, *vector-*, and *segment-based prediction* techniques are proposed [CJNP04, JP07, CJMP05].

The sensors are represented by devices with GPS receivers, and the process and state are vehicle movement and location. Specifically, with *point-based prediction* the server (and an object) predicts that the object remains in the position given to the server in the most recent update. With *vector-based prediction*, the server predicts that the object moves linearly, with constant speed and direction. And in segment-based prediction, it is assumed that an object's movement is restricted to a known road network and it moves along the road segment of the network at constant speed. In all these cases, a new update is generated when the actual position deviates from the predicted one by more than the allowed threshold. In the context of MIRACLE, changing weather conditions or demand and supply in an electricity network are typical continuous processes, which state a need to be tracked. The shared-prediction technique can be applied to make this more efficient. Also, if accuracy guaranties are ignored, predictions of process states can be used to approximate future states thus increasing availability of data in cases when data sources are down.

Another approach to achieve communication-efficient tracking of measurement values can be based on the *Trickle algorithm* [LPCS04, LBC+08]. Trickle is typically used for data propagation and maintenance in wireless sensor networks, but can also potentially be used for the efficient tracking of measurement values. The idea of Trickle is based on the *Polite Gossip* policy, according to which nodes in a wireless sensor network periodically broadcast meta-data to local neighbors, but remain quiet if they have seen only meta-data identical to their own. For example, the meta-data can specify a configuration or version of code or data. According to Trickle, when a node's data does not agree with that of its neighbors, it communicates quickly to resolve the inconsistency. When nodes agree, they slow their communication rate exponentially such that nodes send very few packets in a stable state. A similar approach can be taken in MIRACLE for the tracking of measured values, especially for the type of tracking that involves data pulling, in contrast to the pushing of data.

6.3 Early Aggregation

In general, the term early aggregation is used in the sense of executing any aggregation as early as possible in order to reduced the amount of data to be processed. However, this is a cost-based optimization decision, where we decide when (e.g., at which point of a query execution plan) and how (e.g., over which aggregation dimension) to aggregate. In this context, we mainly distinguish between the two research perspectives of local query processing and distributed query processing, where for the latter the costs of network transfer are taken into account as well.

Note that in the literature the concept of early aggregation is sometimes called pre-aggregation. Unfortunately, there is an ambiguity of this term because it also refers to pre-computation of aggregates in the form of materialized views. For this reason, we exclusively use the term early aggregation in the section.

Early aggregation in DBMS is known as eager group-by [CS94], where the basic idea is to push-down group-by operators and to perform subsequent operations (such as joins) on partitions instead of on individual tuples and therefore, reducing the costs of a query execution plan. In this context, pull-up and push-down techniques exist, where we distinguish between complete [YL95] and partial [Lar02] aggregation. In addition to query processing in DBMS, Ives et al. introduced an adjustable-window early aggregation operator [IHW04] for query processing in EII frameworks (enterprise information integration), where statistics on the base data are unknown. There, early aggregation has been leveraged within an intra-operator adaptive query processing framework.

All these approaches group equal values with regard to given aggregation dimensions. Based on the observation that many applications, required to group similar but not necessarily equal values, the similarity group-by (SGB) [SAA09] was proposed. With regard to electricity data management systems, early aggregation might also rely on similarity such as on time ranges or on similar customer profiles. In addition, this might be a possibility to include uncertain data into early aggregation by using a similarity measure of probability distributions.

In contrast to early aggregation in query execution plans, early aggregation in sensor networks addresses the same objective from a more macroscopic perspective. There, the main goal is to reduce sensor communication by early in-network aggregation in order to achieve energy efficiency. We mainly distinguish two strategies. First, there are tree-based approaches [MFHH02, MFHH03, YG03] that provide exact results in case of no failures but have the disadvantage of being not robust in case of node failures. The problem of a node failure in a tree is that measurements from all subtrees of this node cannot be included into the computation. Second, there are multi-path routing strategies [CLKB04, NGSA04] that are fault-tolerant but might cause more communication efforts and provide only approximate results due to the problem of combining partial results. This led to a hybrid approach [MNG05] that combines the advantages of both. It uses the tree approach for lower hierarchy levels and the multi-path approach for higher hierarchy levels because on higher hierarchy levels a failure has more influence on the overall result than on lower hierarchy levels.

In conclusion, early aggregation is crucial for electricity data management systems in order to meet the mentioned scalability requirements. Especially, techniques from the area of early aggregation in sensor networks such as tree-based or multi-path approaches might be applicable within the MIRACLE scenario as well in order to reduce the amount of transferred data.

6.4 Materialized Views

Materialized views are traditionally used in order to speed up queries by redundantly storing the results of complex queries or subqueries [CKPS95]. This greatly speeds up query processing if these materialized views can be reused by many queries. Essentially, three important aspects must be considered in order to make use of materialized views [GL01]: (1) the design (creation of views), (2) the exploitation (selection of views), and (3) the maintenance of views on updates of the base tables. In the following, we give an overview of the state-of-the-art with regard to these three main aspects.

Typically, materialized views are explicitly created by an administrator or by design advisors for physical design tuning [BN08]. In addition, materialized views can also be created, transparently for the user, by reusing intermediate results [IKNG09] or by exploiting transient views [SV98, ZLFL07]. In order to reduce the number of created views for queries with aggregations, often concepts from the research field of pre-aggregation are applied as well. However, in this context the problem of summarizability in the sense of irregular dimension hierarchies [PJD99] need to be taken into account.

Created materialized views are exploited by determining common subexpressions between queries and the existing materialized views in combination with transparent rewriting of queries. There is plenty of work with regard to this problem of efficient, automated materialized view selection [SDJL96, ACN00, ZCL$^+$00, PH01]. In addition, index structures such as the filter tree have been proposed [GL01] in order to speed up the view matching process.

Matching materialized views is not only important for reusing these views but also for efficient maintenance in case of updates of the underlying base tables. From a macroscopic perspective, the maintenance of materialized views consists of two important aspects, namely the maintenance strategy and the maintenance time aspect. With regard to the applied maintenance strategy, we mainly distinguish between full and incremental maintenance, where for the sake of efficiency, most approaches rely on incremental maintenance [BLT86, GMS93, GL95, MQM97, CR05, LZ07, ZLE07] by computing update deltas. Furthermore, with regard to the time aspect, we distinguish the major categories of eager and deferred maintenance. Eager maintenance [BLT86, GMS93, GL95, CR05, LZ07] refers to applying all updates directly to the materialized views and thus, writing transactions pay for maintenance, while reading queries benefit and obtain up-to-date results. Here, either immediate refresh on each operation or refresh on transaction commit is used. In contrast, deferred maintenance [CGL+96, SBCL00] refers to delaying required refresh operations for a certain period or until a user explicitly triggers refresh. Typically, this deferred maintenance is separated into propagate (compute delta) and apply (incremental maintenance) phases. Thus, updates do not pay for view maintenance but queries might obtain outdated results. In consequence, the lazy view maintenance [ZLE07] tries to combine the advantages of both by deferring maintenance operations and refresh views during free cycles or if a query requests a particular materialized view.

The concept of materialized views is similar to forecast models and consumptions/production schedules in electricity data management systems. For example, estimating the forecast model is similar to the creation of a materialized view and the problem of model selection and model maintenance exists as well. In conclusion, concepts such as incremental maintenance strategies from the area of materialized views can be reused. However, there are major differences with regard to aggregation and disaggregation of forecast models and schedules that require tailor-made maintenance strategies in order to achieve high efficiency.

6.5 Conclusions of Query Optimization

In this section we reviewed some aspects of query optimization, which all can be very applicable in the context of the MIRACLE scenario. In order to meet scalability requirements, we analyzed shared-prediction technique and properties of the Trickle algorithm for efficient tracking of measurements, and early aggregation methods to reduce an amount of processed data. We think that these ideas can be integrated into the MIRACLE infrastructure. Furthermore, we presented materialized views and their application areas in the context of MIRACLE.

7 Existing systems in the energy domain

In this section we will present some of existing systems in the energy domain, specifically, ones used by Energinet.dk (a Danish TSO), a trading platform E-Energy, and dedicated SAP software packages. These concrete solutions are typical examples of the available and currently applied applications in the other energy companies. Moreover, some of the solutions can be used in designing the MIRACLE system.

7.1 Danish Systems

First, we will describe the systems portfolio of the Danish Transmission System Operator (TSO), Energinet.dk. The systems generally have to bridge two aspects of energy production and consumption: a *physical* aspect, where the energy that is fed into the grid must continuously be balanced with the current demand, and a *market* aspect, where the actors providing or consuming power, providing reserve capacity, etc., must be paid, or billed, correctly.

The systems can be classified into Supervisory Control And Data Acquisition (SCADA) systems, systems for operations planning, settlement systems, and systems for long term planning. Additionally, Energinet.dk operates a number of systems related to the infrastructure and emergency management, and a range of administrative systems (primarily based on SAP/R3) similar to most other companies. We will not go into detail with these, since they are not relevant for the MIRACLE project.

SCADA System The SCADA system of Energinet.dk, Delfinen, is used for real-time collection of data from and about the grid, e.g., about production of electricity and the operation of transformer stations. The aim is to supervise and regulate the operation of the grid. As a basis for the operation, the system makes forecasts and performs simulations. This system collects around 600 million rows per day, initially captured at 1 second granularity, followed by later gradually replacing non-recent detail rows with aggregates at first 1 minute, then 5 minute, then 15 minute, and finally 1 hour (or higher), levels. This results in huge volumes of data over time that additionally have the complexity of being *multi-granular*, i.e., the granularity of data items is not uniform. In case of MIRACLE, the similar techniques of gradual aggregation can be used to store data efficiently. On the other hand, Delfinen system operates comparatively smaller amount of data than MIRACLE system, thus new methods suitable for large scale distributed systems must be applied.

Operations Planning System The operations planning system, DPS, is used to handle the actor's *bids* on how much energy, both regular production and so-called spinning power (reserve capacity), they would like to produce in the upcoming 24 hour period. Daily auctions are then held by DPS to select the best bids, and DPS then closes the deals with the bidders. DPS also makes forecasts on production and demand in one-hour intervals over the upcoming 24 hour period. Additionally, DPS simulates the upcoming operations an hour ahead, e.g., shortly after 1PM, DPS starts simulating the upcoming 2PM-3PM operations hour. This is done to check how well the existing 24 hour forecast holds up, to check if the grid can handle the load, and to simulate the effect of possible disruptions (power plant failure, cutting a main wire, etc.). This system collects around 600,000 data rows per day. It is based on MS SQL Server 2005. The MIRACLE infrastructure can benefit from DPS architecture and its forecast validation mechanisms.

Settlement System The billing system, PANDA, is a customized version of a standard billing system for the energy sector, used by many other Danish companies. PANDA knows all the (complicated) rules for settlement, e.g., special subsidies or prices for biomass fuel, older windmills, newer windmills, etc. PANDA knows the power production cost in the hour it was produced, how much should be charged for transmitting it, and how much power is lost in the grid. This is collected as time series with either hourly or quarterly granularity. PANDA then computes the financial consequences of all activities and performs the required settlement. An interesting aspect is that many production figures are first received as a non-validated “best guess”, followed by corrections up to *five years* later, meaning that temporal database concepts such as valid time and transaction time become essential, along with managing the uncertainty in the data; these concepts are also relevant for MIRACLE. Nevertheless, PANDA has several times less clients comparing to (an estimate of) MIRACLE, hence data aggregation methods must be scaled up and large volumes of data must be taken into account. PANDA system collects around 500,000 rows per day. It is based on Oracle 10g Enterprise Edition, and utilizes features such as materialized views and partitioning.

Long Term Planning Systems As part of the TSO role, Energinet.dk must perform long term planning of the demand for energy over the next 5–30 years, and the consequences for grid evolvement. Here, the Netsys system knows the grid topology and is used for simulating a number of scenarios. GIS systems are used to keep track of transformer stations, windmills, and other grid elements. Additionally, in order to regulate the (free) market for electricity, a number of business intelligence (BI) systems are used to report on the state of the market, and check if actors on the market unfairly takes advantage of bottlenecks or a dominating postion. The systems are either self-developed and/or based on SAP/BW and Business Objects. Long term planning is out of scope of the MIRACLE project, thus this system is not directly related to MIRACLE, on the other hand, concepts of monitoring tools and elements are relevant.

7.2 German systems

Here, we describe some business systems implemented by the SAP corporation and also the E-Energy project which aims to promote the expansion of renewable energy sources, to increase energy efficiency and to reduce CO2 emissions.

E-Energy The ”E-Energy” program is a the key component of the ongoing MEREGIO project [MER10] which aims to rise energy efficiency and to reduce climate gas emissions in Germany. The ”E-Energy” system is a trading platform for energy products, system services, and value-added services, designed to integrate energy users and local generators into the market. It supports dynamic tariff models for end consumers and therefore increases flexibility of a consumption. The pilot project was held in Karlsruhe/Stuttgart region with 1000 participants: approx. 800 consumers such as households and small and medium-size companies, approx. 200 local power producers like photo-voltaic plants, micro co-generators etc. Using state-of-the-art IT equipment the participants were integrated into physical and virtual networks using devices for Demand Side and Demand Response Management, remote readout, intelligent multi-utilities meters, etc. The E-Energy system was used to trade electric energy, system services, such as reactive power, control energy, and additional services like energy efficiency consultation. The platform enabled to create incentives for spontaneous response to varying supply and demand, together rising the transparency of pricing mechanisms. MIRACLE has a similar mission, thus some components from ”E-Energy” system may be reused in the MIRACLE.

SAP for Utilities SAP for Utilities is a dedicated solution for companies in the generation, transmission and distribution, retail, and services of energy and water segments. It supports role-specific business processes, optimization of energy portfolios, financial analysis, asset management, and customer relationship management. SAP for Utilities helps to control the supply chain of energy, using analytical tools to forecast and monitor the market, adjust sales prices and enable energy procurement to match demand at lowest possible cost. It also offers smart meter integration which is very relevant for the MIRACLE project. For example, MIRACLE infrastructure may reuse data integration layer, but SAP for Utilities does not offer integration of micro-requests, so this component must be designed and developed separately. The financial analysis and reporting helps companies to decrease operational costs, drive business performance, optimize profits, and control risks. SAP for Utilities supports various asset management processes for enabling equipment reliability, stable environment, zero incident rate, fast commissioning, greater productivity, and operational efficiency. Finally, the software ensures transparent operational chain which optimizes company’s financial results.

SAP Carbon Impact SAP Carbon Impact is an on-demand web based solution for global organizations to help to reduce the enterprise carbon footprint. SAP Carbon Impact offers a single repository of consolidated environmental impact data with direct reporting tools to voluntary or mandatory registries, flexible hierarchies that represent organization’s facilities, departments, product lines, etc, and analytical tools for emissions trending analysis, visualization, and problem identification. The MIRACLE project may benefit from the analytical component by adapting the same techniques for forecasting of energy demand and supply, although more robust solutions for scalability and performance must be implemented.

8 Conclusions

The main challenges of the future electricity network are the active customer involvement and the better integration of renewable energy sources. The MIRACLE project has an objective to achieve both. Its approach is based on the real-time (or almost real-time) operation of the computer infrastructure that processes large number of micro-requests and performs scheduling and forecasting in real-time.

In this document we presented the MIRACLE project, the requirements for the MIRACLE infrastructure, and reviewed the available solutions in order to meet those requirements. We payed special attention to the state-of-the-art research of data collection and integration as well as data analysis.

We presented research results of horizontal and vertical data integration and proposed to make further investigation of materialized integrated systems and real-time data streaming systems. Although they both are relevant in the context of MIRACLE requirements, none of them can be taken as out of the box solution. We analyzed various aspects of the state-of-the-art data exchange and web scale data management systems, and suggested to use a new tailor-made electricity data analysis and collection system. Furthermore, relevant research results on query optimization and distribution techniques were described in the context of the MIRACLE scenario in order to minimize the amount of transfered data and ensure scalability of the system. However we cannot use any of them in MIRACLE directly, thus additional research must be conducted in this field. The types of uncertainties and inconsistencies in the data were identified and methods handling them were reviewed. We also described existing systems in the energy domain and showed their relevance for the MIRACLE project.

The data collection requirements can be satisfied while taking advantage of materialized integrated systems and real-time data streaming management systems. System scalability can be achieved using the same materialized integrated system which is inherently distributed and forms a multi-tier data warehouse infrastructure.

Some techniques from the area of early aggregation in sensor networks can be applied in order to reduce the amount of transferred data, thus reaching a real-time performance of the system.

We identified further research activities based on the current state-of-the-art in data collection and analysis in order to design the future electricity network, since there were no such large scale, highly reliable, precise systems built before.

9 References

References

[AASY97] Divyakant Agrawal, Amr El Abbadi, Ambuj K. Singh, and Tolga Yurek. Efficient view maintenance at data warehouses. In *SIGMOD Conference*, pages 417–427, 1997.

[Aba07] Daniel J. Abadi. Column stores for wide and sparse data. In *CIDR*, pages 292–297, 2007.

[ABPA+09] Azza Abouzeid, Kamil Bajda-Pawlikowski, Daniel J. Abadi, Alexander Rasin, and Avi Silberschatz. HadoopDB: An architectural hybrid of MapReduce and DBMS technologies for analytical workloads. *PVLDB*, 2(1):922–933, 2009.

[ACN00] Sanjay Agrawal, Surajit Chaudhuri, and Vivek R. Narasayya. Automated selection of materialized views and indexes in SQL databases. In *VLDB*, pages 496–505, 2000.

[Age02] International Energy Agency. Renewable energy. `http://www.iea.org/papers/2002/renewable.pdf`, 2002.

[Agg09] Charu C. Aggarwal. *Managing and Mining Uncertain Data*. Springer Publishing Company, Incorporated, 2009.

[AH00] Ron Avnur and Joseph M. Hellerstein. Eddies: Continuously adaptive query processing. In *SIGMOD Conference*, pages 261–272, 2000.

[AMDM07] Daniel J. Abadi, Daniel S. Myers, David J. DeWitt, and Samuel Madden. Materialization strategies in a column-oriented DBMS. In *ICDE*, pages 466–475, 2007.

[AMF06] Daniel J. Abadi, Samuel Madden, and Miguel Ferreira. Integrating compression and execution in column-oriented database systems. In *SIGMOD Conference*, pages 671–682, 2006.

[AMH08] Daniel J. Abadi, Samuel Madden, and Nabil Hachem. Column-stores vs. row-stores: how different are they really? In *SIGMOD Conference*, pages 967–980, 2008.

[ANS05] ANSI/ISO. *ISO/IEC 9075-1:2003 Information technology Database languages SQL*, 2005.

[AW09] Parag Agrawal and Jennifer Widom. Continuous uncertainty in Trio. In *MUD*. Stanford InfoLab, 2009.

[AY09] Charu C. Aggarwal and Philip S. Yu. A survey of uncertain data algorithms and applications. *IEEE Trans. on Knowl. and Data Eng.*, 21(5):609–623, 2009.

[BABO$^+$09] Manish Bhide, Manoj Agarwal, Amir Bar-Or, Sriram Padmanabhan, Srinivas Mittapalli, and Girish Venkatachaliah. XPEDIA: XML processing for data integration. *PVLDB*, 2(2):1330–1341, 2009.

[BB05] Shivnath Babu and Pedro Bizarro. Adaptive query processing in the looking glass. In *CIDR*, pages 238–249, 2005.

[BBS04] Magdalena Balazinska, Hari Balakrishnan, and Michael Stonebraker. Contract-based load management in federated distributed systems. In *NSDI*, pages 197–210, 2004.

[BDJ$^+$05] Doug Burdick, Prasad M. Deshpande, T. S. Jayram, Raghu Ramakrishnan, and Shivakumar Vaithyanathan. OLAP over uncertain and imprecise data. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 970–981. VLDB Endowment, 2005.

[BFG$^+$08] Matthias Brantner, Daniela Florescu, David A. Graf, Donald Kossmann, and Tim Kraska. Building a database on S3. In *SIGMOD*, 2008.

[BFV96] Luc Bouganim, Daniela Florescu, and Patrick Valduriez. Dynamic load balancing in hierarchical parallel database systems. In *VLDB*, pages 436–447, 1996.

[BHF09] Carsten Binnig, Stefan Hildenbrand, and Franz Färber. Dictionary-based order-preserving string compression for main memory column stores. In *SIGMOD Conference*, pages 283–296, 2009.

[BHL10] Matthias Boehm, Dirk Habich, and Wolfgang Lehner. Multi-process optimization via horizontal message queue partitioning. In *ICEIS*, pages 5–14, 2010.

[BHLW08] Matthias Boehm, Dirk Habich, Wolfgang Lehner, and Uwe Wloka. Workload-based optimization of integration processes. In *CIKM*, pages 1479–1480, 2008.

[BHP$^+$09a] Matthias Boehm, Dirk Habich, Steffen Preissler, Wolfgang Lehner, and Uwe Wloka. Cost-based vectorization of instance-based integration processes. In *ADBIS*, pages 253–269, 2009.

[BHP$^+$09b] Matthias Boehm, Dirk Habich, Steffen Preissler, Wolfgang Lehner, and Uwe Wloka. Vectorizing instance-based integration processes. In *ICEIS*, pages 40–52, 2009.

[BHW$^+$07] Matthias Boehm, Dirk Habich, Uwe Wloka, Juergen Bittner, and Wolfgang Lehner. Towards self-optimization of message transformation processes. In *ADBIS Research Communications*, pages 116–125, 2007.

[BJ03] M. H. Böhlen and C. S. Jensen. Temporal data model and query language concepts. *Encyclopedia of Information Systems*, 4:437–453, 2003.

[BLT86] José A. Blakeley, Per-Åke Larson, and Frank Wm. Tompa. Efficiently updating materialized views. In *SIGMOD Conference*, pages 61–71, 1986.

[BMM$^+$04] Shivnath Babu, Rajeev Motwani, Kamesh Munagala, Itaru Nishizawa, and Jennifer Widom. Adaptive ordering of pipelined stream filters. In *SIGMOD Conference*, pages 407–418, 2004.

[BN08] Nicolas Bruno and Rimma V. Nehme. Configuration-parametric query optimization for physical design tuning. In *SIGMOD Conference*, pages 941–952, 2008.

[Bre00] Eric A. Brewer. Towards robust distributed systems. In *PODC*, 2000.

[BSSJ98] Rasa Bliujute, Simonas Saltenis, Giedrius Slivinskas, and Christian S. Jensen. Systematic change management in dimensional data warehousing. In *Proc. 3rd International Baltic Workshop on DB and IS*, 1998.

[BW04] Shivnath Babu and Jennifer Widom. StreaMon: An adaptive engine for stream query processing. In *SIGMOD Conference*, pages 931–932, 2004.

[BWHL08] Matthias Boehm, Uwe Wloka, Dirk Habich, and Wolfgang Lehner. Message indexing for document-oriented integration processes. In *ICEIS*, pages 137–142, 2008.

[BZN05] Peter A. Boncz, Marcin Zukowski, and Niels Nes. MonetDB/X100: Hyper-pipelining query execution. In *CIDR*, pages 225–237, 2005.

[CCMC09] Jinchuan Chen, Reynold Cheng, Mohamed F. Mokbel, and Chi-Yin Chow. Scalable processing of snapshot and continuous nearest-neighbor queries over one-dimensional uncertain data. *VLDB J.*, 18(5):1219–1240, 2009.

[CDG$^+$06] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Michael Burrows, Tushar Chandra, Andrew Fikes, and Robert Gruber. Bigtable: A distributed storage system for structured data (awarded best paper!). In *OSDI*, 2006.

[CDG$^+$08] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Michael Burrows, Tushar Chandra, Andrew Fikes, and Robert E. Gruber. Bigtable: A distributed storage system for structured data. *ACM Trans. Comput. Syst.*, 26(2), 2008.

[CDTW00] Jianjun Chen, David J. DeWitt, Feng Tian, and Yuan Wang. NiagaraCQ: A scalable continuous query system for internet databases. In *SIGMOD Conference*, pages 379–390, 2000.

[CGL+96] Latha S. Colby, Timothy Griffin, Leonid Libkin, Inderpal Singh Mumick, and Howard Trickey. Algorithms for deferred view maintenance. In *SIGMOD Conference*, pages 469–480, 1996.

[CJL+08] Ronnie Chaiken, Bob Jenkins, Per-Åke Larson, Bill Ramsey, Darren Shakib, Simon Weaver, and Jingren Zhou. SCOPE: easy and efficient parallel processing of massive data sets. *PVLDB*, 1(2):1265–1276, 2008.

[CJMP05] Alminas Civilis, Christian S. Jensen, Senior Member, and Stardas Pakalnis. Techniques for efficient road-network-based tracking of moving objects. *IEEE Transactions on Knowledge and Data Engineering*, 17:698–712, 2005.

[CJNP04] Alminas Civilis, Christian S. Jensen, Jovita Nenortaite, and Stardas Pakalnis. Efficient tracking of moving objects with precision guarantees. *Mobile and Ubiquitous Systems, Annual International Conference on*, 0:164–173, 2004.

[CKP03] Reynold Cheng, Dmitri V. Kalashnikov, and Sunil Prabhakar. Evaluating probabilistic queries over imprecise data. In *SIGMOD Conference*, pages 551–562, 2003.

[CKPS95] Surajit Chaudhuri, Ravi Krishnamurthy, Spyros Potamianos, and Kyuseok Shim. Optimizing queries with materialized views. In *ICDE*, pages 190–200, 1995.

[CKSV08] Michael Cammert, Jürgen Krämer, Bernhard Seeger, and Sonny Vaupel. A cost-based approach to adaptive resource management in data stream systems. *IEEE Trans. Knowl. Data Eng.*, 20(2):230–245, 2008.

[CLKB04] Jeffrey Considine, Feifei Li, George Kollios, and John W. Byers. Approximate aggregation techniques for sensor databases. In *ICDE*, pages 449–460, 2004.

[CR05] Songting Chen and Elke A. Rundensteiner. GPIVOT: Efficient incremental maintenance of complex ROLAP views. In *ICDE*, pages 552–563, 2005.

[CRS+08] Brian F. Cooper, Raghu Ramakrishnan, Utkarsh Srivastava, Adam Silberstein, Philip Bohannon, Hans-Arno Jacobsen, Nick Puz, Daniel Weaver, and Ramana Yerneni. PNUTS: Yahoo!'s hosted data serving platform. volume 1, pages 1277–1288, 2008.

[CS94] Surajit Chaudhuri and Kyuseok Shim. Including group-by in query optimization. In *VLDB*, pages 354–366, 1994.

[CXP+04] Reynold Cheng, Yuni Xia, Sunil Prabhakar, Rahul Shah, and Jeffrey Scott Vitter. Efficient indexing methods for probabilistic threshold queries over uncertain data. In *In Proc. VLDB*, pages 876–887, 2004.

[DCSW09] Umeshwar Dayal, Malú Castellanos, Alkis Simitsis, and Kevin Wilkinson. Data integration flows for business intelligence. In *EDBT*, 2009.

[DD99] Ruxandra Domenig and Klaus R. Dittrich. An overview and classification of mediated query systems. *SIGMOD Record*, 28(3):63–72, 1999.

[Dea07] Jeffrey Dean. MapReduce and other building blocks for large-scale distributed systems at Google. In *USENIX Annual Technical Conference*, 2007.

[DG92] David J. DeWitt and Jim Gray. Parallel database systems: The future of high performance database systems. In *Commun. ACM*, volume 35, 1992.

[DG04] Jeffrey Dean and Sanjay Ghemawat. MapReduce: Simplified data processing on large clusters. In *OSDI*, pages 137–150, 2004.

[DHJ$^+$07] Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter Vosshall, and Werner Vogels. Dynamo: amazon's highly available key-value store. In *SOSP*, 2007.

[DRS09] Nilesh Dalvi, Christopher Ré, and Dan Suciu. Probabilistic databases: diamonds in the dirt. *Commun. ACM*, 52(7):86–94, 2009.

[Ene09] Energinet.dk. System plan 2009. `http://www.energinet.dk/NR/rdonlyres/23FA3710-E21D-4611-8A9E-92EFDE43D690/0/Systemplan2009GBweb.pdf`, 2009.

[GL95] Timothy Griffin and Leonid Libkin. Incremental maintenance of views with duplicates. In *SIGMOD Conference*, pages 328–339, 1995.

[GL01] Jonathan Goldstein and Per-Åke Larson. Optimizing queries using materialized views: A practical, scalable solution. In *SIGMOD Conference*, pages 331–342, 2001.

[GL02] Seth Gilbert and Nancy A. Lynch. Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. *SIGACT News*, 33(2), 2002.

[GM95] Ashish Gupta and Inderpal Singh Mumick. Maintenance of materialized views: Problems, techniques, and applications. *IEEE Data Eng. Bull.*, 18(2):3–18, 1995.

[GMS93] Ashish Gupta, Inderpal Singh Mumick, and V. S. Subrahmanian. Maintaining views incrementally. In *SIGMOD Conference*, pages 157–166, 1993.

[GMWD09] Chetan Gupta, Abhay Mehta, Song Wang, and Umeshwar Dayal. Fair, effective, efficient and differentiated scheduling in an enterprise data warehouse. In *EDBT*, pages 696–707, 2009.

[GNC$^+$09] Alan Gates, Olga Natkovich, Shubham Chopra, Pradeep Kamath, Shravan Narayanam, Christopher Olston, Benjamin Reed, Santhosh Srinivasan, and Utkarsh Srivastava. Building a highlevel dataflow system on top of MapReduce: The Pig experience. *PVLDB*, 2(2):1414–1425, 2009.

[GYSD08a] Anastasios Gounaris, Christos Yfoulis, Rizos Sakellariou, and Marios D. Dikaiakos. A control theoretical approach to self-optimizing block transfer in Web service grids. *TAAS*, 3(2), 2008.

[GYSD08b] Anastasios Gounaris, Christos Yfoulis, Rizos Sakellariou, and Marios D. Dikaiakos. Robust runtime optimization of data transfer in queries over web services. In *ICDE*, pages 596–605, 2008.

[GZ86] C. Genest and J. Zidek. *Combining Probability Distributions: A Critique and an Annotated Bibliography*. Statistical Science, vol. 1, 1986.

[HAB$^+$05] Alon Y. Halevy, Naveen Ashish, Dina Bitton, Michael J. Carey, Denise Draper, Jeff Pollock, Arnon Rosenthal, and Vishal Sikka. Enterprise information integration: successes, challenges and controversies. In *SIGMOD*, 2005.

[HLAM06] Stavros Harizopoulos, Velen Liang, Daniel J. Abadi, and Samuel Madden. Performance tradeoffs in read-optimized databases. In *VLDB*, pages 487–498, 2006.

[HML09] Thomas Hornung, Wolfgang May, and Georg Lausen. Process algebra-based query workflows. In *CAiSE*, pages 440–454, 2009.

[HRP$^+$07] Dirk Habich, Sebastian Richly, Steffen Preissler, Mike Grasselt, Wolfgang Lehner, and Albert Maier. BPEL-DT - data-aware extension of BPEL to support data-intensive service applications. In *WEWST*, 2007.

[IBY$^+$07] Michael Isard, Mihai Budiu, Yuan Yu, Andrew Birrell, and Dennis Fetterly. Dryad: distributed data-parallel programs from sequential building blocks. In *EuroSys*, pages 59–72, 2007.

[IHW04] Zachary G. Ives, Alon Y. Halevy, and Daniel S. Weld. Adapting to source properties in processing data integration queries. In *SIGMOD Conference*, pages 395–406, 2004.

[IKM07a] Stratos Idreos, Martin L. Kersten, and Stefan Manegold. Database cracking. In *CIDR*, pages 68–78, 2007.

[IKM07b] Stratos Idreos, Martin L. Kersten, and Stefan Manegold. Updating a cracked database. In *SIGMOD Conference*, pages 413–424, 2007.

[IKM09] Stratos Idreos, Martin L. Kersten, and Stefan Manegold. Self-organizing tuple reconstruction in column-stores. In *SIGMOD Conference*, pages 297–308, 2009.

[IKNG09] Milena Ivanova, Martin L. Kersten, Niels J. Nes, and Romulo Goncalves. An architecture for recycling intermediates in a column-store. In *SIGMOD Conference*, pages 309–320, 2009.

[IR05] Milena Ivanova and Tore Risch. Customizable parallel execution of scientific stream queries. In *VLDB*, pages 157–168, 2005.

[JMSS08a] Theodore Johnson, S. Muthukrishnan, Vladislav Shkapenyuk, and Oliver Spatscheck. Query-aware partitioning for monitoring massive network data streams. In *ICDE*, pages 1528–1530, 2008.

[JMSS08b] Theodore Johnson, S. Muthu Muthukrishnan, Vladislav Shkapenyuk, and Oliver Spatscheck. Query-aware partitioning for monitoring massive network data streams. In *SIGMOD Conference*, pages 1135–1146, 2008.

[JP07] Christian S. Jensen and Stardas Pakalnis. TRAX: real-world tracking of moving objects. In *VLDB '07: Proceedings of the 33rd international conference on Very large data bases*, pages 1362–1365. VLDB Endowment, 2007.

[JSHL02] Vanja Josifovski, Peter M. Schwarz, Laura M. Haas, and Eileen Tien Lin. Garlic: a new flavor of federated query processing for DB2. In *SIGMOD Conference*, pages 524–532, 2002.

[KM05] Martin L. Kersten and Stefan Manegold. Cracking the database store. In *CIDR*, pages 213–224, 2005.

[Kos00] Donald Kossmann. The state of the art in distributed query processing. *ACM Comput. Surv.*, 32(4):422–469, 2000.

[KR02] Ralph Kimball and Margy Ross. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*. John Wiley & Sons, Inc., New York, NY, USA, 2002.

[KS04] Jürgen Krämer and Bernhard Seeger. PIPES - a public infrastructure for processing and exploring streams. In *SIGMOD*, 2004.

[KS09] Jürgen Krämer and Bernhard Seeger. Semantics and implementation of continuous sliding window queries over data streams. *TODS*, 34(1), 2009.

[Lar02] Per-Åke Larson. Data reduction by partial preaggregation. In *ICDE*, pages 706–715, 2002.

[LBC$^+$08] Philip Levis, Eric Brewer, David Culler, David Gay, Samuel Madden, Neil Patel, Joe Polastre, Scott Shenker, Robert Szewczyk, and Alec Woo. The emergence of a networking primitive in wireless sensor networks. *Commun. ACM*, 51(7):99–106, 2008.

[LHY$^+$08] Xiaolei Li, Jiawei Han, Zhijun Yin, Jae-Gil Lee, and Yizhou Sun. Sampling Cube: A Framework for Statistical OLAP over Sampling Data. In *ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'08)*, , Vancouver, Canada, June 2008.

[LLR06] Thomas Legler, Wolfgang Lehner, and Andrew Ross. Data mining with the SAP Netweaver BI Accelerator. In *VLDB*, pages 1059–1068, 2006.

[LÖ09] Ling Liu and M. Tamer Özsu, editors. *Encyclopedia of Database Systems*. Springer US, 2009.

[LPCS04] Philip Levis, Neil Patel, David Culler, and Scott Shenker. Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks. In *In Proceedings of the First USENIX/ACM Symposium on Networked Systems Design and Implementation (NSDI*, pages 15–28, 2004.

[LZ05] Haibo Li and Dechen Zhan. Workflow timed critical path optimization. *Nature and Science*, 3(2), 2005.

[LZ07] Per-Åke Larson and Jingren Zhou. Efficient maintenance of materialized outer-join views. In *ICDE*, pages 56–65, 2007.

[LZJ$^+$05] Bin Liu, Yali Zhu, Mariana Jbantova, Bradley Momberger, and Elke A. Rundensteiner. A dynamically adaptive distributed system for processing complex continuous queries. In *VLDB*, pages 1338–1341, 2005.

[MBK00] Stefan Manegold, Peter A. Boncz, and Martin L. Kersten. Optimizing database architecture for the new bottleneck: memory access. *VLDB J.*, 9(3):231–246, 2000.

[Mel09] Sergey Melnik. The frontiers of data programmability. In *BTW*, pages 5–6, 2009.

[MER10] The `MEREGIO` project. `http://www.meregio.de`, June 2010.

[MF04] Roger MacNicol and Blaine French. Sybase IQ Multiplex - designed for analytics. In *VLDB*, pages 1227–1230, 2004.

[MFHH02] Samuel Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. TAG: A tiny aggregation service for ad-hoc sensor networks. In *OSDI*, 2002.

[MFHH03] Samuel Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. The design of an acquisitional query processor for sensor networks. In *SIGMOD Conference*, pages 491–502, 2003.

[MG97] Simon Marvin and Simon Guy. Smart metering technologies and privatised utilities. *Local Economy*, 12:119–132, 1997.

[MNG05] Amit Manjhi, Suman Nath, and Phillip B. Gibbons. Tributaries and deltas: Efficient and robust aggregation in sensor network streams. In *SIGMOD Conference*, pages 287–298, 2005.

[Mot97] Amihai Motro. *Uncertainty Management in Information Systems: From Needs to Solutions*. Kluwer Academic Publishers, Norwell, MA, USA, 1997.

[MQM97] Inderpal Singh Mumick, Dallan Quass, and Barinderpal Singh Mumick. Maintenance of data cubes and summary tables in a warehouse. In *SIGMOD Conference*, pages 100–111, 1997.

[MTM08] Ramon Mata-Toledo and Morgan Monger. Implementing a temporal data management system within oracle. *J. Comput. Small Coll.*, 23(3):76–81, 2008.

[MWGW10] Joel Makower, Clint Wilder, Dexter Gauntlett, and Trevor Winnie. Clean energy trends 2010. Technical report, Clean Edge, 2010.

[NGSA04] Suman Nath, Phillip B. Gibbons, Srinivasan Seshan, and Zachary R. Anderson. Synopsis diffusion for robust aggregation in sensor networks. In *SenSys*, pages 250–262, 2004.

[NRB09] Rimma V. Nehme, Elke A. Rundensteiner, and Elisa Bertino. Self-tuning query mesh for adaptive multi-route query processing. In *EDBT*, pages 803–814, 2009.

[O'C08] William O'Connell. Extreme streaming: business optimization driving algorithmic challenges. In *SIGMOD*, 2008.

[ORS$^+$08] Christopher Olston, Benjamin Reed, Utkarsh Srivastava, Ravi Kumar, and Andrew Tomkins. Pig latin: a not-so-foreign language for data processing. In *SIGMOD Conference*, pages 1099–1110, 2008.

[ÖV96] M. Tamer Özsu and Patrick Valduriez. Distributed and parallel database systems. *ACM Comput. Surv.*, 28(1):125–128, 1996.

[ÖV97] M. Tamer Özsu and Patrick Valduriez. Distributed and parallel database systems. *The Computer Science and Engineering Handbook*, pages 1093–1111, 1997.

[ÖV99] M. Tamer Özsu and Patrick Valduriez. *Principles of Distributed Database Systems*. 1999. Second Edition.

[PDGQ05] Rob Pike, Sean Dorward, Robert Griesemer, and Sean Quinlan. Interpreting the data: Parallel analysis with Sawzall. *Scientific Programming*, 13(4):277–298, 2005.

[PH01] Rachel Pottinger and Alon Y. Halevy. Minicon: A scalable algorithm for answering queries using views. *VLDB J.*, 10(2-3):182–198, 2001.

[PJD99] Torben Bach Pedersen, Christian S. Jensen, and Curtis E. Dyreson. Extending practical pre-aggregation in on-line analytical processing. In *VLDB*, pages 663–674, 1999.

[PPR+09] Andrew Pavlo, Erik Paulson, Alexander Rasin, Daniel J. Abadi, David J. DeWitt, Samuel Madden, and Michael Stonebraker. A comparison of approaches to large-scale data analysis. In *SIGMOD Conference*, pages 165–178, 2009.

[PVHL09] Steffen Preissler, Hannes Voigt, Dirk Habich, and Wolfgang Lehner. Stream-based web service invocation. In *BTW*, pages 407–417, 2009.

[RDS+04] Elke A. Rundensteiner, Luping Ding, Timothy M. Sutherland, Yali Zhu, Bradford Pielech, and Nishant Mehta. CAPE: Continuous query engine with heterogeneous-grained adaptivity. In *VLDB*, pages 1353–1356, 2004.

[RM95] Erhard Rahm and Robert Marek. Dynamic multi-resource load balancing in parallel database systems. In *VLDB*, pages 395–406, 1995.

[SAA09] Yasin N. Silva, Walid G. Aref, and Mohamed H. Ali. Similarity group-by. In *ICDE*, pages 904–915, 2009.

[SAB+05] Michael Stonebraker, Daniel J. Abadi, Adam Batkin, Xuedong Chen, Mitch Cherniack, Miguel Ferreira, Edmond Lau, Amerson Lin, Samuel Madden, Elizabeth J. O'Neil, Patrick E. O'Neil, Alex Rasin, Nga Tran, and Stanley B. Zdonik. C-Store: A column-oriented DBMS. In *VLDB*, pages 553–564, 2005.

[Sas06] C. Sasse. Electricity networks of the future. In *Power Engineering Society General Meeting, 2006. IEEE*, pages 7–7, 2006.

[SBCL00] Kenneth Salem, Kevin S. Beyer, Roberta Cochrane, and Bruce G. Lindsay. How to roll a join: Asynchronous incremental view maintenance. In *SIGMOD Conference*, pages 129–140, 2000.

[SBL04] Sven Schmidt, Henrike Berthold, and Wolfgang Lehner. QStream: Deterministic querying of data streams. In *VLDB*, pages 1365–1368, 2004.

[Sc05] Michael Stonebraker and Ugur Çetintemel. ”one size fits all”: An idea whose time has come and gone (abstract). In *ICDE*, pages 2–11, 2005.

[SCS+08] Adam Silberstein, Brian F. Cooper, Utkarsh Srivastava, Erik Vee, Ramana Yerneni, and Raghu Ramakrishnan. Efficient bulk insertion into a distributed ordered table. In *SIGMOD*, 2008.

[SDJL96] Divesh Srivastava, Shaul Dar, H. V. Jagadish, and Alon Y. Levy. Answering queries with aggregation using views. In *VLDB*, pages 318–329, 1996.

[SHCF03] Mehul A. Shah, Joseph M. Hellerstein, Sirish Chandrasekaran, and Michael J. Franklin. Flux: An adaptive partitioning operator for continuous query systems. In *ICDE*, pages 25–36, 2003.

[SJ06] Albrecht Schmidt and Christian S. Jensen. Efficient maintenance of ephemeral data. In *DASFAA*, pages 141–155, 2006.

[SJS06] Albrecht Schmidt, Christian S. Jensen, and Simonas Saltenis. Expiration times for data management. *Data Engineering, International Conference on*, 0:36, 2006.

[SLSL05] Sven Schmidt, Thomas Legler, Sebastian Schär, and Wolfgang Lehner. Robust real-time query processing with QStream. In *VLDB*, pages 1299–1302, 2005.

[SMA$^+$07] Michael Stonebraker, Samuel Madden, Daniel J. Abadi, Stavros Harizopoulos, Nabil Hachem, and Pat Helland. The end of an architectural era (it's time for a complete rewrite). In *VLDB*, pages 1150–1160, 2007.

[SMM$^+$08] Sarvjeet Singh, Chris Mayfield, Sagar Mittal, Sunil Prabhakar, Susanne E. Hambrusch, and Rahul Shah. Orion 2.0: native support for uncertain data. In *SIGMOD Conference*, pages 1239–1242, 2008.

[SMS$^+$08] Sarvjeet Singh, Chris Mayfield, Rahul Shah, Sunil Prabhakar, Susanne E. Hambrusch, Jennifer Neville, and Reynold Cheng. Database support for probabilistic attributes and tuples. In *ICDE*, pages 1053–1061, 2008.

[SMWM06] Utkarsh Srivastava, Kamesh Munagala, Jennifer Widom, and Rajeev Motwani. Query optimization over web services. In *VLDB*, pages 355–366, 2006.

[Sto02] Michael Stonebraker. Too much middleware. *SIGMOD Record*, 31(1), 2002.

[SV98] Subbu N. Subramanian and Shivakumar Venkataraman. Cost-based optimization of decision support queries using transient views. In *SIGMOD Conference*, pages 319–330, 1998.

[SVS05a] Alkis Simitsis, Panos Vassiliadis, and Timos K. Sellis. Optimizing ETL processes in data warehouses. In *ICDE*, pages 564–575, 2005.

[SVS05b] Alkis Simitsis, Panos Vassiliadis, and Timos K. Sellis. State-space optimization of ETL workflows. *IEEE Trans. Knowl. Data Eng.*, 17(10):1404–1419, 2005.

[SWCD09] Alkis Simitsis, Kevin Wilkinson, Malú Castellanos, and Umeshwar Dayal. QoX-driven ETL design: reducing the cost of ETL consulting engagements. In *SIGMOD Conference*, pages 953–960, 2009.

[SWDC10] Alkis Simitsis, Kevin Wilkinson, Umeshwar Dayal, and Malú Castellanos. Optimizing ETL workflows for fault-tolerance. In *ICDE*, pages 385–396, 2010.

[TBL09] Maik Thiele, Andreas Bader, and Wolfgang Lehner. Multi-objective scheduling for real-time data warehouses. *Computer Science - R&D*, 24(3):137–151, 2009.

[TDP06] Igor Timko, Curtis E. Dyreson, and Torben Bach Pedersen. Pre-aggregation with probability distributions. In *DOLAP '06: Proceedings of the 9th ACM international workshop on Data warehousing and OLAP*, pages 35–42, New York, NY, USA, 2006. ACM.

[TFL09] Maik Thiele, Ulrike Fischer, and Wolfgang Lehner. Partition-based workload scheduling in living data warehouse environments. *Inf. Syst.*, 34(4-5):382–399, 2009.

[THL10] Jacopo Torriti, Mohamed G. Hassan, and Matthew Leach. Demand response experience in europe: Policies, programmes and implementation. *Energy*, 35(4):1575 – 1583, 2010. Demand Response Resources: the US and International Experience, Demand Response Resources: the US and International Experience.

[TPL08] Christian Thomsen, Torben Bach Pedersen, and Wolfgang Lehner. RiTE: Providing on-demand data for right-time data warehousing. In *ICDE*, pages 456–465, 2008.

[TPL+10] Thanh T.L. Tran, Liping Peng, Boduo Li, Yanlei Diao, and Anna Liu. PODS: a new model and processing algorithms for uncertain data streams. In *SIGMOD '10: Proceedings of the 2010 international conference on Management of data*, pages 159–170, New York, NY, USA, 2010. ACM.

[TTT+05] Igor Timko, Copyright C Igor Timko, Author(s Igor Timko, Curtis E. Dyreson, Curtis E. Dyreson, Curtis E. Dyreson, Torben Bach Pedersen, Torben Bach Pedersen, and Torben Bach Pedersen. Probabilistic data modeling and querying for location-based data warehouses. In *In Proceedings of 17th International Scientific and Statistical Database Management Conference (SSDBM)*, pages 273–282, 2005.

[TVS07] Vasiliki Tziovara, Panos Vassiliadis, and Alkis Simitsis. Deciding the physical implementation of ETL workflows. In *DOLAP*, pages 49–56, 2007.

[VSES08] Marko Vrhovnik, Oliver Suhre, Stephan Ewen, and Holger Schwarz. PGM/F: A framework for the optimization of data processing in business processes. In *ICDE*, pages 1584–1587, 2008.

[VSS+07] Marko Vrhovnik, Holger Schwarz, Oliver Suhre, Bernhard Mitschang, Volker Markl, Albert Maier, and Tobias Kraft. An approach to optimize data processing in business processes. In *VLDB*, pages 615–626, 2007.

[WK10] Richard Winter and Pekka Kostamaa. Large scale data warehousing: Trends and observations. In *ICDE*, 2010.

[WPB+09] Thomas Willhalm, Nicolae Popovici, Yazan Boshmaf, Hasso Plattner, Alexander Zeier, and Jan Schaffner. SIMD-Scan: Ultra fast in-memory table scan using on-chip vector processing units. *PVLDB*, 2(1):385–394, 2009.

[YB08] Qi Yu and Athman Bouguettaya. Framework for web service query algebra and optimization. *TWEB*, 2(1), 2008.

[YG03] Yong Yao and Johannes Gehrke. Query processing in sensor networks. In *CIDR*, 2003.

[YIF+08] Yuan Yu, Michael Isard, Dennis Fetterly, Mihai Budiu, Úlfar Erlingsson, Pradeep Kumar Gunda, and Jon Currey. DryadLINQ: A system for general-purpose distributed data-parallel computing using a high-level language. In *OSDI*, pages 1–14, 2008.

[YL95] Weipeng P. Yan and Per-Åke Larson. Eager aggregation and lazy aggregation. In *VLDB*, pages 345–357, 1995.

[ZCC10] Yinuo Zhang, Reynold Cheng, and Jinchuan Chen. Evaluating continuous probabilistic queries over imprecise sensor data. *Database Systems for Advanced Applications*, 5981:535–549, 2010.

[ZCL+00] Markos Zaharioudakis, Roberta Cochrane, George Lapis, Hamid Pirahesh, and Monica Urata. Answering complex SQL queries using automatic summary tables. In *SIGMOD Conference*, pages 105–116, 2000.

[ZLC10] Jingren Zhou, Per-Åke Larson, and Ronnie Chaiken. Incorporating partitioning and parallel plans into the SCOPE optimizer. In *ICDE*, pages 1060–1071, 2010.

[ZLE07] Jingren Zhou, Per-Åke Larson, and Hicham G. Elmongui. Lazy maintenance of materialized views. In *VLDB*, pages 231–242, 2007.

[ZLFL07] Jingren Zhou, Per-Åke Larson, Johann Christoph Freytag, and Wolfgang Lehner. Efficient exploitation of similar subexpressions for query processing. In *SIGMOD Conference*, pages 533–544, 2007.

[ZRH04] Yali Zhu, Elke A. Rundensteiner, and George T. Heineman. Dynamic plan migration for continuous queries over data streams. In *SIGMOD Conference*, pages 431–442, 2004.