D-DAL

Data: Dynamics Algorithmics and Logics

Team leader: Sylvain Salvati
Inria research center: Inria centre at the University of Lille

Field: Perception, Cognition and Interaction

Theme: Data and Knowledge Representation and Processing

In partnership with Centre de Recherche en Informatique, Signal et Automatique de Lille (CRIStAL)

Genealogy: this team is a follow-up of LINKS

Summary: D-DAL scientific endeavour is to *extract value from data with high efficiency.* Extracting value means that D-DAL will explore new kinds of queries that allow to gather more information from data. It also means that D-DAL will work with large data displaying very different properties. Data will have many forms (graphs, relations, trees, texts, etc.), different features (incomplete, probabilistic, etc.), submitted to many changes. Achieving efficiency within these many contexts requires that D-DAL carefully studies in which cases query languages can be efficiently processed. It also requires that D-DAL tries to exploit hardware capabilities as much as possible. D-DAL will combine, via compilation methods, efficient query resolution algorithms to construct high throughput data processing pipelines. The approach of D-DAL takes its roots in formal methods, more specifically: logic, algorithms, automata, algebra.

Proposal outline:

Last updated: 06/03/2026

Contact: sylvain.salvati@univ-lille.fr

1 Team members

D-DAL: *Data Dynamic Algorithmic and Logic* is a proposal of INRIA project that aims to pursue research at the frontier of foundational research and applications in databases.

The conception of D-DAL takes place in the context of the end of the LINKS INRIA project which after 12 years arrived at its administrative end. Compared to the members LINKS, the members of D-DAL are slightly different. In terms of departures, Joachim Niehren (DR Inria, HDR) has left the team to join BioComputing, Sławek Staworko (Associate professor Univ. Lille, HDR) has left to work in the private sector, and Florent Capelli (Associate professor Univ. Lille) has left to join the CRIL laboratory at Lens in a CPJ position. On the other hand, D-DAL will welcome Antoine Amarilli (previously Associate professor at Télécom Paris, HDR) as a new member. Sylvain Salvati will take the lead of the team.

Our new focus is about the design and implementation of efficient query languages for various contexts: various kinds of data, various knowledge about data, dynamic settings, various ways of serving data, etc. A singularity of D-DAL is that it aims to integrate query resolution algorithms with compilation methods so as to construct efficient data processing pipelines. The skills of D-DAL's members cover all the aspects of this positioning: finite state machine, functional programming, algorithmic, logic, etc.

Permanent members

- Antoine Amarilli, ARP Inria, HDR, `https://a3nm.net/`
- Iovka Boneva, Associate Professor Univ. Lille, `https://pro.univ-lille.fr/iovka-boneva`
- Aurélien Lemay, Associate Professor Univ. Lille, HDR, `https://pro.univ-lille.fr/aurelien-lemay`
- Mikaël Monet, CRCN Inria, `https://mikael-monet.net/`
- Charles Paperman, Associate Professor Univ. Lille, HDR, `https://paperman.name/`
- Sylvain Salvati, Professor Univ. Lille, HDR, team leader, `https://pro.univ-lille.fr/sylvain-salvati`

Non permanent members

- Sophie Tison, Professor Emeritus Univ. Lille, HDR, `https://pro.univ-lille.fr/sophie-tison`

PhD students

- Bastien Degardins, Funding ANR JCJC Find-RNA, started October 2024, supervised by Camille Marchet (BONSAI team) and Charles Paperman
- Oliver Irwin, Funding ANR JCJC KODA, started December 2022, supervised by Florent Capelli and Sylvain Salvati, `https://www.irwin.sh/`
- Ahmet Ucan, Funding 50% Univ. Lille 50% Region, starting January 2025, supervised by Antoine Amarilli, Mikaël Monet, and Sylvain Salvati

2 Context, overall objectives and challenges

A major and early success story of computer science, combining theoretical foundations and efficient implementations, is the development of *relational databases*. Relational databases have now been deployed in essentially all sectors of activity, from large multi-server clusters to phones and embedded devices. An important factor of this success is the abstraction lying at the core of relational databases: they combine a simple data model with a declarative language (SQL) to write user queries. These queries are then transparently translated to optimized execution plans taking advantage of the schema (the data model) and statistics of the data. This abstracts away all internal details that would be unpleasant to the user: how data is concretely stored, the shape of the actual execution plan, low level optimizations, the orchestration of data streams that compute intermediate query results, concurrent accesses, and so on.

Building on this foundation, the field of data management has evolved in two antagonistic directions over the past decade. One first direction is that the scope of data uses in applications has significantly expanded. Data is used to train statistical models; they are the basis of key decisions that have a strong impact on people and on society, and must therefore be *explainable*. The data that we store may be *updated* often, which makes

it necessary to efficiently recompute the query answers. In particular, the data may arrive in *streams*, and we may have to compute query answers on the fly, even before we have read the entire data stream. The number of query answers to produce may be so large that we need to produce it in streaming, or to access it implicitly, instead of querying it in full. The data is *heterogeneous*: it does not necessarily follow a fixed schema like the relational model traditionally assumes, and it may feature non-relational data such as textual documents, tree-shaped data (such as XML and JSON records), and so on. In particular, data may be stored as a graph, without a fixed schema; it may then be interrogated with queries featuring *recursion*, e.g., asking for pairs of reachable nodes. There may be *uncertainty* on the data stored by such systems, making it difficult to compute accurate answers. Last, some queries need to be evaluated across datasets from different sources, e.g., large open datasets from multiple public organizations. In such settings, data is typically not stored within one single system, but *integrated* across different systems. Further, beyond traditional query evaluation, the data can be transformed, combined, and post-processed. This can be done with *procedural languages* within SQL databases (stored functions and procedures), or with *domain specific languages* such as LINQ that are used to querying data within the programming language C#. Thus, many data processing frameworks are actually libraries that can be seen as query languages embedded within a programming language; conversely, some query languages such as XPath 3.0 for XML documents or `jq` for JSON documents now embed limited programming languages so as to make them more expressive. These new developments are blurring the line between querying and programming, and are giving up on the idea that data manipulation needs should be expressed declaratively.

Faced with the growth of these new applications, the scope of the SQL standard has significantly expanded to accommodate some (but not all) of the corresponding needs. Recent versions of SQL support for recursive queries, incremental view maintenance and triggers, stored procedures, queries over strings, embedded XML and JSON documents and query over them, along with naive forms of uncertainty and incompleteness (through NULLs). Most recently, the graph query language GQL has been published as part of SQL. These features are standardized and supported by relational engines, though the coverage and efficiency of these features may vary across implementations. Further, for some of these tasks, such as data integration and post-processing, an entire ecosystem of solutions has developed on top of relational databases.

The second direction is that the data has increased in volume. Some organizations now want to store much larger volumes of data, e.g., by extracting it from the Web, collecting scientific data, or running services for an unprecedented number of users. Traditional relational databases, with their support of the full SQL language, may not be suited to such large data volumes. For this reason, in the 2000s, new "NoSQL" database paradigms started to become popular. These systems compromise on some aspects of SQL, such as its expressiveness (i.e., supporting less expressive queries), or its formal guarantees (e.g., distributed consistency), to achieve better performance. Highly-optimized systems now exist for specific tasks, such as key-value stored, document-oriented databases, etc.

To summarize, new data management applications now ask for two conflicting aspects: the support for *more expressive queries* (going beyond the more arcane features of SQL); while offering *more efficient querying* over very large quantities of data. These features should be offered while retaining the guiding principle of data management systems, namely, going from a high-level declarative language that describes user needs, down to low-level execution plans that implement them over data representations. Parts of this translation can be *data-independent*, using, e.g., static analysis; or they can be *data-dependent*, using, e.g., logical constraints or statistics of the database.

The main goal of D-DAL (D-DAL is pronouced as *Dédale*, the French pronunciation of Daedalus) is to find paths to extract meaning from the maze of data. In other words, D-DAL's scientific objective consists in overcoming the limits of expressivity and efficiency of programs dealing with data. This global objective gives rise to two main challenges. A challenge with a large perspective of *extracting value from data*, and a practical and pragmatic challenge of *processing large data efficiently*.

Challenge 1: Extracting value from data

The first challenge that we will address is that of supporting more expressive ways to interrogate data, so as to make sense of complex data. We will address several aspects of this challenge. First, the data can be uncertain[1], imprecise, or probabilistic. Second, it can be unstructured, or with an unknown structure that must be discovered or formalized. Third, we may need to extend query evaluation to compute complex information, in particular explanations, but also provenance annotations, complex aggregation values, or complex results, in a principled manner.

[1]We use the term uncertain data as it is used in the field of databases. It refers to data which is either unknown or that contains some probabilistic aspects.

Dealing with uncertain data. Data in real life is rarely certain: it can be subject to human error when manually collected, generated from physical sensors having imperfect precision, or extracted from sources of various quality from the Web via natural language processing (NLP) or machine learning techniques, that are often themselves of a probabilistic nature. The traditional way of dealing with this uncertainty is to simply ignore it, e.g., by removing all rows containing NULL values in a database before evaluating a query. This naive approach can however lead to incorrect or incomplete answers and is thus not always desirable. The challenge is then to design methods and algorithms that are able to take this uncertainty into account in a principled and efficient way, i.e., by reasoning over the possible completions of the data in a logical fashion, or by reasoning quantitatively over the possible worlds of probabilistic data to determine which query results are likely.

Finding structure in data. Relational databases assume that the data is strictly structured by a fixed *data schema*: this assumption is central to the query language, and to efficient algorithms to select query execution plans. However, when data comes from various sources, or when it is updated with new kinds of information, then relational databases are inadequate: they require time-consuming engineering tasks to refactor data schema to reflect the new needs. By contrast, the model of *graph databases* is more flexible and has become increasingly popular for applications. As efficiency comes at a cost, flexibility also has a cost, and data may display irregularities and peculiarities that make it harder to process. We focus on two main research directions to explore on data graphs.

The first research direction is to develop ways to express restrictions on the shape of graph data, via flexible notions of schemas. Schemas make it possible to trade some of the flexibility of the graph model with the efficiency of the relational model: they make it possible to accommodate diverse data representations in a controlled manner, while disallowing specific kinds of irregularities, and while guiding query evaluation algorithms to optimize query evaluation plans with the information given by the schema.

The second approach consists in analyzing data in order to reveal its structure. Through comprehensive data examination and modelling techniques, we aim to automatically discover the underlying patterns and organization within the graph. These methods could lead to the inference of a schema that will be exploited by other algorithms to exploit directly the data, to transform it into a usable form, or to directly help inference of queries or transformation.

Explaining query results. The increasing usage of computing in every aspect of modern society has naturally raised a need for transparency in algorithms. In some scenarios, having the answer to a query is not enough and one also wants some kind of *explanation* on how the result was derived, or wants to sort the possible results according to some importance score.

In particular, in contexts where decisions are made on the basis of data that can impact people's lives, like in medicine, or that can impact society, like in policy making, it is important not to just query data, but also to understand why the answers are obtained, or even understand how pieces of data weigh in the answers using certain aggregates like Shapley values. Indeed data is imperfect and query explanation is a way to deal with this imperfection. These concerns are not confined to the database community: the problem arises in all data-driven computations. Most notably, with the rise of AI, *explainable AI* has emerged as an important scientific domain.

From a technical point of view, query explanability can be understood more generally as building complex query answers that keep track of the operations performed during the evaluation, amounting to a form of user-defined aggregation; it can also be understood in the abstract setting of *semiring provenance*. All these new tasks can incur an increase in the time to answer queries, and thus call for the development of new notions of explanations and of new efficient algorithms.

Challenge 2: Processing large data efficiently

The standard of SQL is continuously growing. Now, it encompasses a larger number of data formats and ways to query these formats: recursive queries, XML and Xquery, JSON, and more recently property graphs. Most of these extensions of the standard are encoded in the relational model, sometimes in very simple manners. For example, JSON documents are in general[2] stored as strings. Instead of attacking the problem of handling massive data by trying to treat all kinds of formats and queries at once, like the SQL standard does, we prefer to concentrate on particular fragments and propose efficient analyses, optimization methods and algorithms for these fragments. When the number of query results is very large, specific methods are needed to construct them. For instance, streaming the results one by one dilutes the computational effort over time. In situations when queries are repeated, another strategy may be to store the results of the query and update it along with

[2]A notable exception is PostgreSQL, which proposes the type JSONB that allows for better indexation and avoids the cost of parsing.

the data. We shall, in some cases, develop high throughput prototypes for these specific query fragments. In the end, the composition of these tools should result in highly efficient data processing pipelines.

Elegant and efficient query languages. As data comes in many forms (text, JSON documents, graphs, relational databases, ...), query languages are highly diverse (regular expressions, JSONpath, Xpath, SQL, the numerous NOSQL querying APIs, ...). An important challenge is to understand what is computationally feasible and what is not. Knowing which *fragments* of the query languages can benefit from efficient evaluation will possibly allow us to develop specific tools to handle them. We are going to focus on particular aspects of expressive queries that go beyond the traditional relational model: recursion, uncertainty handling, and support for embedded programming. For these features, we aim at studying which aspects of queries make evaluation more or less efficient. This will result in the design of fast algorithms for some families of queries and proofs of computational hardness for others. Beyond these results, for practically useful families of queries, we shall develop domain specific languages and prototypes that take advantage of the algorithms that we will have designed.

Beyond this theoretical endeavour, we also wish to explore how efficient query evaluation algorithms can be articulated with modern hardware platforms. Indeed, hardware now offers increasingly parallel, concurrent or distributed primitives. We believe that one of the keys to efficient query evaluation on large data is to align novel algorithmic methods with modern hardware.

Last, in settings where efficient query evaluation remains out of reach, we intend to study approximation algorithms, to obtain approximate answers to queries with reasonable time bounds. Approximation can be understood in a logical sense (i.e., principled overapproximations or underapproximations of query results), in a quantitative sense (i.e., approximate numerical or probabilistic query answers), or in a practical sense (i.e., approximations that are validated by an experimental approach).

Enumeration and incremental algorithms. Querying data is only one aspect of data processing. Indeed, when looking at entire data processing pipelines, we see that results are often combined with other results coming from other data sources, and can be further processed by programs. This poses new research questions: instead of optimizing query evaluation as a monolithic task, we can study how to globally optimize query evaluation pipelines by studying the interplay between the evaluation of several queries.

In this light, we intend to study two common algorithmic frameworks to deal with large amounts of data, which are well-suited to orchestrate the simultaneous evaluations of many queries in a pipeline. These are *enumeration algorithms* and *incremental algorithms*.

The principle of enumeration algorithms is the following: when the number of query results is huge, it is neither feasible nor realistic to produce the entire collection of results in answer to a user query. Instead, it is preferable to *enumerate* the results in streaming, one after the other, while minimizing the *delay* between two successive answers. Evaluating queries by enumerating results offers two potential advantages. First, downstream applications in the pipeline can access query results more quickly, as soon as they are computed. Second, outputting results one by one might drastically reduce the amount of memory needed to perform the computation.

The principle of incremental algorithms is that queries that are repeatedly executed can be drastically accelerated by memorizing previous results. The idea of incremental algorithms is to avoid the recomputation of all the results; this is usually done by designing data structures that can efficiently be maintained when the data is updated and that can serve to compute quickly the new query result.

Designing query execution engines. Once query results from various data sources are enumerated either by an enumeration algorithm or from data structures that store them, one needs to orchestrate and synchronize the processing of these results. In this context results are abstracted as streams that are combined together.

Techniques pertaining to programming languages are well suited to work on streams. Streams of data and their combinations are well studied in the context of synchronous and reactive programming. Generalizations have been also studied by the community of functional programming languages. Bringing these methods to deal with queries and more broadly to data processing can have a strong impact on the efficiency of query execution engines.

The relation with programming languages and thus to compilation naturally leans towards trying to exploit hardware optimally to improve data throughput. In particular, exploiting all the levels of parallelization: SIMD[3] instructions, the several cores that are inside CPUs, GPUs, or in a distributed context.

[3]SIMD stands for *Single Instruction Multiple Data* and are CPU instructions that allow to operate at the same time on several machine words with the same CPU instruction like bitwise operations, additions, substractions, etc.

Finally another challenge is to develop methods at the frontier of enumeration and incremental algorithms, i.e., to develop data structures that support the enumeration of query results, and that can also be updated efficiently when the data itself is updated, so as to start again the enumeration of the (possibly new) query results.

3 Scientific objectives and methodology

Our research organisation addresses the two scientific challenges that motivate the projet D-DAL, *extracting value from data* and *processing large data efficiently*, with three axes that correspond to different level of abstractions with respect to these challenges.

The first axis, the *basement*, is about *circuits* which are a transverse to the two challenges. Circuits form a handy abstraction for representing data and understanding it structure but also a proxy that allows to study the complexity of queries and also to understand how to take the most out of modern CPUs.

The second axis, the *first floor*, is about querying. It builds upon what is produced in the *basement* and is intended to develop efficient querying algorithms and eventually querying programs. This axis is concerned with explaining query results, studying fragments of querying languages, dynamics aspects of querying and streaming. The *first floor* builds upon the basement, circuits play an important role in explaining query results and various aggregation mechanisms as the provide the right structure to represent concisely query results and then execute efficiently computations on these results. Circuits also provide the frontier of the feasible and the efficient about queries and they will allow us to focus on the right query fragments. Finally, circuits are the most basic streaming algorithms we are going to study and implement. The study of streaming methods will allow us to compose efficiently these basic streaming algorithms and enable the creation of efficient querying pipelines.

The third axis, the *roof*, is about proposing and exploiting information about data. The *roof* has two *slopes*, one is about proposing and studying schemas for graphs and the other is about taking into account knowledge about data so as to accelerate querying. The first slope is concerned with the schema language ShEx which has been developed within Links (the team from which D-DAL originates) and D-DAL will pursue this effort as it gives us concrete problems in relation to data for query optimization exploiting knowledge about data. The second slope is precisely about taking into account knowledge about data when answering queries. This information should make algorithms more efficient and also to work with uncertain data more consistently. We will work on the specific information coming from ShEx schemas, but also with more general settings. Overall, the *roof* axis stands upon the second axis as it aims at refining the work done on querying and applying it to settings where we can take advantage from extra hypotheses in order to lower the complexity.

In a a nutshell, the three axes build a coherent architecture which aims to work out the maze of data. We here give a summary of their contributions to the two challenges:

Extracting value from data

Basement Circuits are natural and concise representations of data, be it raw data or answers to queries. With tools coming from knowledge compilation, they reveal the inherent structure of data, which can be used independently from the queries that we intend to evaluate over it.

First floor Querying is the basic way to extract information from data, which can then be refined into value. We plan to explore various querying fragments that enrich the state of the art of querying, including expressive queries featuring recursion, aggregation, and uncertainty handling. It will enable the extraction of richer information.

Roof The development of ShEx is a concrete proposal for describing data in ways that allow to extract value from graph data by taking logical constraints into account. Furthermore, taking advantage from extra information for querying, such as deduction rules and integrity constraints, will expand the capabilities of querying.

Efficient data processing

Basement Circuits can play the role of abstract proxy towards exploiting modern CPUs to process data efficiently. Furthermore, they also play an important role in the study of computational complexity, and in particular *low* complexity classes, or classes allowing parallel computation. Such classes will allow us to describe the landscape of queries that can be solved very efficiently on large data.

First floor Query fragments, and algorithms to process them efficiently, are the central focus of this axis. In particular, the dynamic aspects of queries will be studied in two aspects. First by proposing enumeration and incremental algorithms for queries. Second by studying streaming and the composition of streams.

Roof Taking advantage of information about data and developing ShEx are central to exploiting real world data, which often comes with additional information such as schemas. We will refine querying algorithms to make them more efficient with extra hypotheses, in particular to choose query evaluation plans based on statistics of the data.

Our overall goal is to design highly efficient querying languages in various contexts (data formats, uncertainty, etc.) and the tools that allow to combine them into efficient data processing programs. D-DAL is on the ridge between expressiveness and efficiency. We have structured our project into tree axes that are detailed in the rest of this section. We summarize their objectives below according to the two challenges we attack.

Extracting value from data

Short term We are going to extend queries and data representation in several directions. First we will work on enhancing circuit classes by negation, and we will study generalizations of structural constraints that allow for efficient counting algorithms and hence handle quantitative uncertainty. Second we will also incorporate probabilities in ShEx and work on queries on graphs with constraints. We will also study which logical constraints can be used to reason on data, and which counterfactual problems (e.g., resilience, Shapley values) can benefit from our existing expertise on related tasks. These short term objectives build on our previous work and work towards more expressive query fragments.

Medium term We plan to take a more general view on negation within circuits, in particular work more thoroughly in connection with knowledge compilation. One goal is to obtain a better understanding of the probabilistic setting specifically. Another goal is to work on open-world query answering in the presence of uncertainty and of recursion. We also plan to study the lifting of constraints of data graphs to views on graphs. This medium-term objectives aim at having a better qualitative understanding of data.

Long term We plan to build a unifying view of the relation between circuit representations and querying problems. This will provide us with an integrated view of how data can be represented. We also plan to work on relating aggregation and uncertainty problems with formalisms used for constraint satisfaction problems, to explore how such approaches can lead to new structural algorithms for query evaluation. We also hope that such approaches can help connect the multiple communities that propose constraint languages for data, in particular graph-shaped data. Finally, we plan to have ShEx be standardized by the IEEE and have a set of open software tools for it.

Efficient data processing

Short term We are going to consolidate our team's first works which relate circuits to efficient implementations based on SIMD instructions. We shall also work on dynamic and incremental algorithms for simple data like texts and trees, again building on our existing research providing first results in these directions. We will also study efficient enumeration of query results that respect a specific order (the order is a good property for further processing), and connect this with efficient dynamic algorithms. All these works give basic building blocks for streaming algorithms, that we plan to orchestrate via work on simple programs called *transducers*.

Medium term We shall build a circuit-based compilation chain, which goes from query fragments to efficient code, generalizing previous work to transducers. We will also study how the compilation can be doubled with efficient query plan selection at runtime, leveraging the efficient data structures that we will have designed by then. We will also explore how dynamic and incremental algorithms can accommodate expressive features such as recursion and aggregation without compromising the effectiveness of query evaluation, and identify tractability assumptions that can be leveraged to that end. In particular, in connection with enumeration algorithms, we shall work on how to efficiently compose data processes, in particular connecting enumeration algorithms that produce sequence of diffs with incremental algorithms that maintain a query result over evolving data.

Long term We shall propose an end-to-end compilation framework for efficient query compilation and the orchestration of heterogeneous systems, including expressive tasks such as reasoning and optimization. We shall take the direction of probabilistic programming to sacrifice precision in favor of efficiency for the evaluation of queries, guided by a fine-grained theoretical understanding of the possible approximation tradeoffs. We will also undertake this precise complexity study in the dynamic and incremental settings as well as for enumeration problems, again aiming at a full understanding of the landscape of possible complexity regimes.

3.1 The Basement: Circuits for Data Manipulation

Large data poses challenges that are both theoretical and practical. D-DAL intends to use *circuits* as a common abstraction to address several of these problems. Circuits are an abstract object that can be approached from various angles, corresponding to different research communities. First, circuits are studied from the *knoweldge compilation* perspective, an AI field which studies classes of Boolean circuits and which has connections to data management [4]. Second, circuits are studied as a *compilation target*, this time to translate programs to abstract representations which ensure that they can be evaluated in a massively parallel or vectorized way. Third, circuits can be studied from a fully theoretical angle, in the field of *circuit complexity theory*. We now present these axes.

Studying circuits from Knowledge Compilation. Participants: Antoine Amarilli, Charles Paperman, Mikaël Monet, Sylvain Salvati

Knowledge compilation [14] is a subfield of AI that studies different ways to represent Boolean functions, in particular restricted classes of *Boolean circuits*. It studies tradeoffs between *conciseness* and *tractability*: which representations can be used for which computational tasks (e.g., counting satisfying assignments), and when can we efficiently translate from one representation to another. It turns out that knowledge compilation is a useful tool to solve many types of database tasks [4], such as factorized representations of query answers, enumeration algorithms, provenance computation, and uncertainty management (in particular on probabilistic data).

We wish to pursue this study of knowledge compilation to databases and query evaluation, and also investigate knowledge compilation in itself to achieve a better understanding of these circuit classes.

→ *Short term.* In the short term, we will explore how specific circuit classes can be used for efficient evaluation of particular queries, across various tasks such as enumeration and ranked access. The PhD thesis of Oliver Irwin is set in this context: it is co-directed by Florent Capelli from CRIL and Sylvain Salvati. It studies how to efficiently find query answers using circuits for queries that may feature negation, and how to reinterpret efficient query evaluation algorithms in a knowledge compilation perspective. We further intend to pursue our study of query evaluation on probabilistic databases via circuit classes, understanding the power and limitations of this method.

→ *Medium term.* One longer-term challenge to be attacked by D-DAL is to understand the power of *negation* in circuit formalisms that allow tractable counting. Indeed, for the circuit classes commonly used in knowledge compilation [1], the use of negation is highly restricted. However, it is not well-understood whether the expressive power of such circuits is extended when we allow arbitrary negation. This question is purely about knowledge compilation circuit classes, but it also has consequences for questions in database theory about tractable query evaluation on probabilistic databases [8]. Another challenge is whether knowledge compilation classes can also capture more tractable enumeration tasks, in particular those involving negations (e.g., enumerating objects which *do not have* a given property); these would be necessary for a knowledge-compilation-based proof of tractable enumeration for some formalisms such as first-order logic on restricted graphs [27].

→ *Long term.* A far-reaching question in knowledge compilation and database theory concerns the relationship between two kinds of results: algorithms (namely, efficient query evaluation algorithms, and computational complexity hardness results), and tractable circuits (namely, efficient compilation algorithms, and lower bounds on the size of tractable circuits). One long-term research direction of the D-DAL team would be to unify these settings. For instance, we could try to understand which hardness results on query evaluation can imply circuit lower bounds, e.g., in the context of recursive queries [5]; and when tractable algorithms can be translated to circuit classes, subsuming the inclusion-exclusion conjecture [8].

Another long-term goal of D-DAL could be to consolidate existing results about knowledge compilation. The current reference on the topic is the seminal work of Darwiche and Marquis [14], but it dates back to 2002. We have started a survey of knowledge compilation in [1], but far more efforts would be needed in order to survey all known results for knowledge compilation circuit classes.

Circuits as a compilation target for parallelism and vectorization.
Participants: Charles Paperman, Sylvain Salvati

Most of the data exchanged on the web is now represented as JSON files which are text documents (other document formats are also in use such as XML or YAML documents). This is a consequence of the fact that many data engine now operate and communicate with these kinds of data. Extracting information and answering queries from textual documents has become a pervasive task when dealing with large amount of data. However with the growth of data that comes with larger and larger documents, processing textual documents has become an important bottleneck in data processing.

For the moment this bottleneck is addressed first by speeding up the translation of the documents into in memory data structures on which queries are then executed. This translation, also called *parsing*, is accelerated via low level optimizations. The quickest implementations have required expert programmers and complex costly developments. They leverage specific CPU instructions known as *vectorial instructions*. The diversity of CPU architectures require the production of highly specialized code for each of them. This work is worth doing since the resulting code is often accelerated by one or two orders of magnitude compared to the non-vectorial variant.

D-DAL aims at developing tools that automatize these optimization and reduce the cost of vectorizing programs that handle textual data. We will not only optimize parsing, but a combination of parsing with query resolution. The PhD thesis of Claire Soyez-Martin [30, 25] (conducted under the supervision of Charles Paperman and Sylvain Salvati) advocates that such automations can be done via a notion of *vectorial circuits*. Our plan is two fold: first study the compilation of expressive queries into vectorial circuits and then build dedicated compilation schemes for vectorial circuits that takes benefits of vectorial instructions.

→ *Short term.* In continuation of the work engaged with the PhD thesis of Claire Soyez-Martin, we are going to develop further the model of vectorial circuits. The aim is to obtain a cost model for these objects that reflects the efficiency of their compilation to CPU and vectorial instructions. We will also identify application domains that can benefit from our methods. We have already identified several possibilities: biological sequence analysis, network monitoring, pervasive streaming tasks in relation with data (e.g. utf8 validation, JSON processing, etc.)

→ *Medium term.* We plan to develop a compilation chain that will contain two steps. The first step will consist in sets of compilers from query fragments to vectorial circuits. The second will be a set compilers from vectorial circuits to efficient vectorized hardware code. At first we will aim at X86_64 architectures. This compilation pipeline will be the result of ground work conducted in algebraic automata theory.

→ *Long term.* We plan to extend our work to other hardware components. One of them is the RiscV platform which has recently been endowed with an interesting set of vectorial instructions. Another one is the processing in memory units (PIM) that provide a highly parallel architecture that could benefit from the relationship between automata theory and circuits.

Foundations of circuits: circuit complexity, algebra and abstract machines.
Participants: Antoine Amarilli, Charles Paperman, Mikaël Monet, Sylvain Salvati

Our work on circuits takes its root in deep connections between logic, formal language and algebra. These deep connections are at the heart of finite state verification techniques that have been flourishing in the last decades. We exploit these connections in a different way. For us logic is connected to queries as logical languages naturally express properties on object which in turn coincides with the use of declarative languages to extract information from data. Moreover, logical formulae on data of bounded size can be represented as circuits which extract information from data. Languages or language transformations are a natural counter part of logical formulae. In general, they can be represented as finite state machines. Finally, algebra (monoids, semigroups, forest algebras) naturally expresses the invariants of these machines. Classes of algebra, computation power of finite state machines, complexity of circuits are then deeply connected. The classification of algebras with respect to their properties often coincides with natural classes of machines and of circuits. Algebras form a useful mediation between machines and circuits, that is, between sequential devices and parallel ones. More generally, algebras formalize the structure of the information that needs to be captured when processing queries. It thus plays a central role in many of the applications D-DAL aims at developing. We plan to develop our understanding of these deep connections in relation with querying with applications in many objectives of D-DAL, most notably: vectorization, incremental maintenance, enumeration, transducers, etc.

→ *Short term.* In connection to vectorization, we shall explore particular classes of circuits and describe the classes of languages they correspond to. We shall also develop algebraic tools for forest algebras to enhance the stream processing on tree structured documents. In particular, we wish to extend the expressiveness of the queries that our software `rsonpath` can solve. Concerning incremental maintenance and enumeration, we shall workout some algebraic invariants on trees that will allow us to extend the methods that we have developed for strings.

→ *Medium term.* The algebraic work on data (strings, trees, graphs, ...) can be extended to more dynamic objects like transducers and even to programs via denotational semantics. In general, queries not only check properties of data, but also transform data. Extending algebraic methods to such dynamic setting will allow us to abstract data processing pipelines and open the way to further optimizations.

→ *Long term.* In the long term, we aim at developing a coherent corpus of algebraic methods for trees and graphs that lend themselves to classify queries on structured documents and graphs. We wish also to attack difficult classification problems that remain open on circuit complexity for classes of regular languages. This ground work shall set the stage for future applications similar to `rsonpath`.

3.2 First Floor: Logic and Query Evaluation

We now present scientific objectives which operate at a higher level, and deal with query evaluation itself, rather than with the tools presented in the previous section which are ways to implement query evaluation.

Provenance, explanation, aggregation, counting, uncertainty, probabilistic data, Shapley, approximation algorithms.
Participants: Antoine Amarilli, Mikaël Monet

As already mentioned in Section 2, querying data often goes beyond merely computing a set of answers and returning them. First, data is often uncertain by nature (e.g., probabilistic or missing values). Then, algorithms must be developed that take into account this uncertainty, by considering all possible states of the data; which turns out to often be intractable. In this context, approximate query answers (e.g., approximating the probability that a probabilistic database satisfies a query) can be employed to lower the complexity. Second, one sometimes wishes to *explain* the query results, using for instance the notion of *provenance* or using specific scoring mechanisms such as the *Shapley value* or the notion of *resilience* of a query result.

→ *Short term.* It is known that when probabilistic data has bounded *treewidth*, then probabilistic query evaluation (PQE) can be solved efficiently for many queries. The PhD thesis of Ahmet Ucan, that should start soon, aims at exploring restrictions that are less limiting than bounded treewidth but could still ensure tractable PQE. One idea would be to design width measures that could be parameterized by the queries. Ideally, we would also obtain lower bounds for specific width measures and queries as it was done in [7].

→ *Medium term.* Another research direction would be to explore the tractability of diverse counterfactual scoring mechanisms, such as Shapley values as done in [15, 23], or the resilience [18] (intuitively representing how "robust" a query answer is, by determing how many input tuples should be removed from the database so that the specific answer no longer exists) for queries featuring recursion.

→ *Long term.* An intriguing and potentially fruitful research direction would be to connect the kind of counting problems studied in database theory (such as PQE or Shapley values) to other well-established counting problems from other fields such as counting versions of *constraint satisfaction problems* (CSPs) [16, 22] or Holant problems and hollographic reduction [13]. We will also strive to provide a toolbox to use these algorithms in practice, leveraging for instance the existing implementation of ProvSQL [29], an open-source PostgreSQL extension that computes (between other things) the Boolean provenance of a query over a database.

Dynamic data: incremental and dynamic algorithms and connections with dynamic algorithms.
Participants: Antoine Amarilli, Charles Paperman, Mikaël Monet, Sylvain Salvati
In many query evaluation scenarios, the data to interrogate is regularly updated. When re-evaluating queries multiple times over different versions of the same data, it is more efficient to build index data structures which can maintain the results of the query under updates, rather than re-evaluating the query every time from scratch. This problem of query evaluation over *dynamic data* has been studied in many different contexts. The D-DAL team will focus on two broad kinds of contexts: *tree-shaped data* on the one hand, which includes texts, trees, genomic sequences, etc.; and *unrestricted data*, namely graph databases and general relational databases.

→ *Short term.* In the short term, we intend to focus on the first application setting of texts and trees, and pursue our ongoing work in these settings. While the main results in the area have already been proven, some work remains in expanding the frontiers of what can be done efficiently over dynamic text and trees: can we maintain data structures for in-order enumeration? in which orders? with which complexity? One important direction which remains open is that of showing improved bounds on query evaluation which depends on the specific query (rather than for a general query class). From our initial results for regular queries on words [6], one especially promising ongoing effort is to extend these results to trees. Another is to extend such results to enumeration data structures. In both cases, significant algebraic groundwork is required.

→ *Medium term.* We then intend to focus on broader directions. One of them is to support *more expressive updates*: the bulk of the literature on dynamic data focuses on single-character or single-tuple changes, but real updates are often more expressive. On text, we want to be able to support *copy-and-paste* updates, which move an entire part of the text to a different point; or *search-and-replace* updates. On relational databases, we want to develop incremental algorithms that cope with updates which are themselves expressed in SQL, i.e., with UPDATE statements. Another direction is to move to the study of incremental maintenance for *more expressive queries*, in particular those featuring recursion or aggregation. For instance, can we characterize how efficiently we can maintain the results of Datalog queries, with fine bounds that depend on the specific Datalog query which is posed? Which assumptions on the underlying data can be used to achieve tractable algorithms?

→ *Long term.* A more open direction for query evaluation over dynamic data would be to achieve precise bounds on the complexity of maintaining each individual query by charting the space of possible tradeoffs across various performance measures: what is the time needed to handle updates to the data; what is the delay between successive results? how much memory usage is required to store the data structure? which updates are supported? In this light, one especially challenging obstacle is the need for suitable lower bound techniques, in particular unconditional lower bounds, which are sometimes achievable depending on the model studied. Another general question is to connect more broadly to the literature studying data structures, in which questions about mutable data have been investigated for decades. We could for instance leverage results on maintaining search trees balanced under updates, or algorithms on *dynamic graphs* [21] which have been studied in the algorithms community (especially algorithms to dynamically maintain the connectivity of directed graphs).

Efficient evaluation: streaming and parallelism.
Participants: Aurélien Lemay, Charles Paperman, Sylvain Salvati

Efficiency in data processing mostly comes from streaming data. This means that data is passed from small processing units to others so as to produce an output. Certain operations cause serious synchronization problems: sorting data, eliminating duplicates, combining streams ... They block the processing and cause sometimes important lags. Sticking to functional programming, D-DAL will develop streaming data processing programs by using static analysis, speculative computations (to prepare results when lags are unavoidable), develop stream fusion compilation methods so as to obtain efficient streaming algorithms. Techniques coming from synchronous programming will be considered, in particular in connection with the leveraging of fine-grained parallelism for low level streaming. Also, in the medium term, we shall explore how to parallelize query executions taking advantage of static analyses of the functional code.

→ *Short term.* Part of the effort will concern very simple classes of functional programs known as *transducers*. D-DAL will implement libraries for manipulating transducers, minimizing them and composing them. Several extensions of the models of the literature are being proposed by the team and will benefit from these implementations. Of particular interest to us are higher-order transducers which form an elegant generalization of many transducers from the literature. We shall work on compiling them for streaming documents.

→ *Medium term.* Orchestrating programs that stream data is challenging as they may work at different paces and as communications between them do not have a negligible cost. We will work on this orchestration problem in connection with programming languages theory, most notably with synchronous and functional programming languages. Indeed, when we have compiled programs with our method, we shall be able to efficiently compose them. Many technical methods for achieving this have been developed for these programming languages.

→ *Long term.* In the long run, we will take stock on the work of the community of functional probabilistic programming. Indeed, when working with huge data, approximate querying becomes hardly avoidable and thus extending our methods to sampling data with various probabilistic distributions becomes necessary. In particular, we shall see how we implement probabilistic querying by means of transducers.

Enumerating the results of recursive queries.
Participants: Antoine Amarilli, Mikaël Monet, Charles Paperman, Sylvain Salvati

One important challenge in modern database theory is efficient *enumeration algorithms* [28], i.e., efficiently producing query results one after the other. This is related to query evaluation in streaming, but with a key difference: here we are looking for algorithms which *produce* a stream of results, instead of *consuming* a stream of input data. Database theory has studied which queries can efficiently be evaluated, but an especially challenging topic is to understand under which assumptions it can be done for *recursive queries*, e.g., Datalog queries. This task can in particular be studied over restricted formalisms of input data, such as textual data. This is the focus of *document spanners* [17], which are a declarative way to specify how to extract matches of queries over textual documents. There, recursivity is needed to express long-distance dependencies (e.g., extract all phone numbers that are between two specific delimiters, no matter how far).

→ *Short term.* One immediate research objective is to study which enumeration tasks for (recursive) queries can be efficiently solved, in particular when considering variations on the standard setting of enumeration algorithms. For instance, in many applications, we want to produce results in a specific *order*; to *directly access* individual results by their offset; or to efficiently *test* whether a candidate result is actually a solution, or finding out its position in the enumeration order. We intend to pursue our ongoing work in such directions [2].

→ *Medium term.* A medium-term research question is the combination of enumeration algorithms with the previous axis of handling *dynamic data*. Previous work by members of the team has already studied how enumeration algorithms could be applied to dynamic data via knowledge compilation methods [3]. However, we are still missing an understanding of how efficient such algorithms can be made, especially if we want query-dependent algorithms. This would probably require an algebraic theory of queries which can return results (to cover document spanners), in particular understanding their connection to transducers.

A related medium-term research question is the idea of enumerating query results *via updates*, i.e., producing each query results not from scratch but by editing a previous result. Team members have started investigating this question in a specific context [9], but this enumeration framework could be fruitful in many other settings. It also has an intriguing connection to the question of query evaluation over dynamic data, because the two problems are related: query evaluation on dynamic data takes a stream of changes as input and maintain query results, whereas enumerating query results via updates will produce a stream of changes to go over all results to a query.

→ *Long term.* Most research about enumeration algorithms for recursive queries using circuit methods focus on the case of textual data (e.g., document spanners) and tree-shaped data. This leaves open the setting of more general data, in particular graph data. Of course, the task has already been studied, in particular in the comparatively simple setting of regular path queries [24]. But we are still missing an understanding of those enumeration tasks that can be solved efficiently in more expressive query languages (e.g., Datalog queries) and depending on the various possible semantics (e.g., returning simple paths, trails, shortest paths, or arbitrary walks). As many of these tasks will be intractable, efficient algorithms will probably also need to identify conditions on the input data that are sufficient to ensure tractability.

Another wide open direction for long-term research in enumeration algorithms concerns lower bounds. Indeed, many lower bounds are conditional, and they usually do not apply to enumeration but to related problems (e.g., checking whether a solution exists, or materializing all solutions). An especially difficult challenge would be to develop lower bound methods that show the intractability of efficient enumeration per se.

3.3 The roof: Knowledge on Data

Optimization for schemas for graphs.
Participants: Aurélien Lemay, Charles Paperman, Iovka Boneva

The role of schemas for graph data is a bit different than schemas for relational data. The latter are prescriptive (i.e. data is organized exactly like this) and are an essential ingredient for static optimization of queries. The former are *de facto* often descriptive (i.e. data should look more or less like this). Such looseness of schemas is unavoidable, and even desirable, for big data stores or knowledge bases which are constantly evolving, fed with data from numerous sources that can be incomplete, inconsistent and contain errors.

D-DAL aims at providing tools for increasing the usefulness of such approximate schemas in the context of dynamically evolving data. This line of research continues our work of the formalization, the standardization the

development and the tooling of the shape expressions language (ShEx) for RDF graphs, and aims at extending on some of our previous results and approaches to property graphs and PG-Schemas.

→ *Short term.* A first short term objective is to enrich graph schema languages such as ShEx with probabilistic information indicating the degree of compliance to the actual data. We will aim at a practical extension with limited expressiveness in order to keep feasible the associated computational tasks (data validation, schema extraction) while still helping experts in assessing the quality of a data set, detecting errors and monitoring the evolution of data. This approach will be built on our previous work in discovering schemas from existing data, while extending it to property graph schemas.

→ *Medium term.* As all software, data-centric applications constantly evolve, either to match new use cases, or to exploit new data sources, or both. This most often requires to transform the data model in order to match the new processing needs. As a mid-term objective we plan to investigate algorithms for incremental and conjoint data and schema evolution, based on logic tools such as mappings and the chase.

Although there exist a number of native graph data stores, lots of data is effectively stored in relational or other data stores (e.g., Wikibase for wikidata) and is only exposed as a graph to the user. There might also be the case that data from streams is processed and integrated to a knowledge graph. The original data can contain schema information or satisfy some constraints. As a mid-term objective, we plan to investigate lifting such constraints from the underlying data to the graph view. We will focus on useful and feasible sub-classes.

→ *Long term.* Finally, we will continue our participation in Shape Expressions Working group which defines the ShEx language, currently in the process of standardization at IEEE. We will also continue to participate in the implementation of a validator for ShEx and related tools as open software.

Static analysis with constraints (open-world query answering, query rewriting under constraints, consistency, certain answers).
Participants: Antoine Amarilli, Aurélien Lemay, Iovka Boneva, Mikaël Monet, Sylvain Salvati, Sophie Tison

Contrary to the *closed world assumption* of traditional database research, the absence of a piece of information often does not mean that this piece of information is false, but rather that we do not know its status; this is called the *open world assumption*. In this scenario, the data is typically enriched by *constraints* that it should satisfy, such as key, cardinality or equality constraints. Several questions arise when querying data under the open world assumption. Is the data consistent with the constraints we know about it? Taking into account the constraints and the fact that the information is incomplete, what degree of certainty can we have about the answer set of queries? Is it possible to take advantage of the constraints so as to execute simpler queries (e.g., using rewriting techniques)? Moreover, it may not be possible to know which constraints apply to the data at stake, in which case it can then be desirable to infer knowledge using machine learning methods. D-DAL will apply the querying methods it develops to that setting so as to produce added values from knowledge bases.

→ *Short term.* We will pursue work we have engaged with the PhD of Lily Gallois [19] on regular path queries (RPQs) and word constraints on graph databases. Word constraints are the simplest kind of constraints that can be put on a graph database; they seem to naturally connect with rewrite systems. As we showed in [26], this relation is however more complicated that expected. We are going to study further the classes of constraints that enable reasoning about RPQs and their decidability. We will also explore the problems of certain query answering and query rewriting in this setting. The aim is to propose efficient query resolution algorithms from which we can build more complex queries, in particular queries with aggregation.

We are also going to work on schema and constraints inference for graph databases. We have skills in symbolic learning methods which have proved useful in extracting information from the web. Facing large graphs, these methods will have difficulties to scale, moreover they are often not robust to noise. Machine learning methods, typically neural networks, are more immune to this problem. We shall develop a hybrid approach: the method that we envision consists in learning structures with neural networks and then extracting the information that is useful in the context of data processing by means of symbolic techniques.

→ *Medium term.* The D-DAL team intends to study several medium-term problems related to open-world query answering and reasoning under rules. One especially interesting question is how the notion of *circuits* can be leveraged to achieve tractable algorithms for these tasks. The hope is that circuits can be a versatile tool unlocking many results that would otherwise need to be shown independently: supporting uncertain facts (and uncertain rules); providing provenance information to capture how a specific fact is derived and explain

to the user why a query result is true (given the data and the rules); giving factorized representations of a possibly large number of query answers. At a technical level, a broad question is how forward-chaining deduction techniques such as the *chase* can be extended to create a circuit representation of the derivations. This is challenging because the chase, unlike circuits, may represent some cyclic dependencies, which may cause convergence issues depending on the semiring in which the computation is performed. Another question is how this can be combined with backward-chaining methods such as query rewriting.

→ *Long term.* In the long run, one unifying challenge of the ontology-mediated query answering community is the question of unifying different logical frameworks. Indeed, while query languages are relatively standard, rules can nowadays be specified in many different incompatible ways: description logics, existential rules, graph patterns, database constraint languages such as TGDs, etc. It is challenging to understand the connections between these settings and difficult to conceive systems that could orchestrate query evaluation across back-ends using different query formalisms. Another more algorithmic challenge concerns the complexity of ontology-mediated queries: while there are many results showing complexity upper and lower bounds for general classes (e.g., guarded TGDs), it is much more challenging to try to identify precise complexities of *individual queries*, in the spirit of what is done for less expressive queries such as conjunctive queries. Thus, the D-DAL team aims at making progress towards this very ambitious theoretical goal.

4 Application domains

The project of the D-DAL team aims at efficiently extracting information from large datasets. We aim to process data in all the diversity of its representations, and to take into account its quality, including questions of uncertainty and probability. Our approach is to develop theoretical results that can later be implemented into concrete programs. As such, our project relates to several INRIA strategic axes:

Data Science and Artificial Intelligence: The work of our team will study how to efficiently produce data for other downstream applications that analyze it or aggregate it from various sources. This is key in analysing data where large chunks of data are processed so produce statistics and other kinds of aggregates. Apart from efficient data processing, the longer terms objectives of the project that aim to incorporate randomized programming hinges towards wider possible scenarios in data science. Similarly, machine learning algorithms are fed with data. These algorithms require efficient data manipulation and also a control over data quality. Our work on uncertainty and provenance are useful in this setting. Finally, the older sense of AI (symbolic AI) is connected to D-DAL through the study of circuits and knowledge compilation. We hope that these connection will reveal fruitful and that our contributions will also have impact in this field.

Software and Hardware We intend to develop query execution engines, querying platform and schemas validators for graphs. Moreover, D-DAL will also have interests in query languages, their extensions, and their execution. These topics make us have close interest in compilation techniques and programming language methods. We adopt an integrating view of query processing that embeds computational capabilities. This leads to see data processing and its interactions with other software differently, delegating more and more computation to query engines.

D-DAL will develop compilation methods for various types of query, taking advantage of parallel hardware capabilities. In particular, we intend to tools to analyze queries so as to detect opportunities for parallel execution. Models of abstract circuits will allow us to harness the power of modern hardware, in particular making better use of SIMD instructions. We also plan to work on RISC V architectures and processing in memory units (PIM). Databases have always had a tradition of exploiting hardware capabilities. However, with open source architectures such as RISC V, the particular needs of databases may be addressed by particular instructions. We hope that our work can have influence on RISC V architectures and that specific hardware design for data processing can emerge from our research.

Databases Our view of databases is different from the integrating view that SQL standardization gives. While SQL continues to integrate in a top-down manner more and more querying capabilities under the same software, the approach of D-DAL is to combine with optimized streaming techniques, in a bottom-up fashion, well-defined and limited query languages for which we have identified highly efficient algorithms. Though more difficult to approach than the SQL view, our view may be more adapted for data processing in contexts where data is scattered within files in different formats or is too large for database management systems.

5 Software

D-DAL is aiming to validate and make impact through software. We are going to develop software along two axes. The first axis concerns RDF graphs and the shape expressions language (ShEx). The second axis concerns platforms for efficient querying of data.

ShEx The shape expressions language has been developed in the long run by the INRIA team LINKS. We plan to pursue the development of ShEx and related tools. We are mainly interested in *ShEx validator* for apache-jena. For this development, we bring our expertise both on the formal definition of ShEx, its specification, and on the validation algorithms. The rest of the software is developed by the team of engineers[4] already involved in this project. Our interest in this project is that it allows us to reach a large community of users.

Querying platform We shall develop platforms of tools that compile queries to efficient programs or support the execution of powerful algorithms on large data. We have already developed such tools and envision their further development.

A first tool is `rsonpath` which focuses on a restricted set of JSONpath queries and allows to process them by streaming extremely fast with a small memory imprint. The reason why this software is so efficient is that it benefits from deep theoretical work and from hardware acceleration. The theoretical work was necessary to determine the family of queries which could be efficiently processed by streaming [11]. In particular, subtle algebraic properties are at the root of the small memory imprint avoiding to memorize information about the tree structure being explored. Concerning hardware acceleration, it is based on vectorization (i.e. SIMD instructions) [20] and the careful articulation of theoretical properties of queries with low-level code.

A second tool is `NetworkDisk` that is aiming at porting the graph algorithms implemented in the `Python` library `NetworkX` to graphs that are stored in databases. This work is now put in application in the analysis of biological sequences. Batien Desgardin is starting as PhD thesis with Charles Paperman and within the team Bonsai in order to put `NetworkDisk` at work in this setting.

These two examples give us a direction for the future development of software. In particular they show that we are able to bring deep theoretical results to very efficient software. Our main line of development will be concerned with hardware accelerated query algorithms in the same vein as `rsonpath`. We shall develop tools that make the necessary algebraic analyses required to analyse queries. As explained Section 3, we shall develop a compiler from vector circuits to low level code that harness vectorized instructions. We shall also develop tools for describing and compiling streaming processes. Another set of tools will be devoted to orchestrating these processes. These tools will take stock on the development of synchronous languages. In the end, we will have developed a set of composable tools that will allow to build complex data processing pipelines. As a guideline and integrating direction we shall consider to extend `rsonpath` to more general queries that fall into the scope of `jq`. This application will play an integrating role for several directions of D-DAL. Indeed, `jq` uses recursive queries described by means of regular expressions, translates documents into other documents representing the answer of the query, contains higher-order feature so as to compute complex aggregation... this gives applications to our work on query evaluations, transducers, higher-order transducers, circuits, algebras. Our strategy to follow `jq`'s settings will allow us to focus on a clear goals and have an impact on data processes that are based on JSON documents which become pervasive.

shex-jena `https://github.com/fhircat/jena`, Apache Licence 2.0

Support of Shape Expression Language 2.1. This ShEx validator is currently under development as a fork of the apache-jena suite, to which it should be merged after reaching the expected maturity. It extends the algorithms from ShExJava to the new constructs of the ShEx language. D-DAL is mainly involved in the algorithmic core of the validator.

- **Software Family**: Transfer software.
- Audience: `community`. This is the intended audience, for now the software is under development.
- Evolution and maintenance: `lts` As part of the apache-jena suite the effort on long term support will be shared with the community.
- Duration of the Development (Duration): 1 year

NetworkDisk `https://networkdisk.inria.fr/`, MIT.

NetworkDisk provides a way to manipulate graphs on disk. The goal is to be as much as possible compatible

[4] These engineers work on this open source project as part of their jobs.

with (Di)Graph objects of the NetworkX Python package but lifting memory requirement and providing persistence of the Graph.

From a research perspective, NetworkDisk is a PoC of query engine for graph database built on the top of query rewriting techniques. While it is usable, it remains at the prototype phase. Still, some teams in bioinformatics and Software Engineering use it to store and manipulate graph data in a friendly ecosystem.

- **Software Family**: Software as a Vehicle for Research.
- Audience: `partners`
- Evolution and maintenance: `lts`.
- Duration of the Development (Duration): 4 years.

rsonpath `https://github.com/rsonquery/rsonpath`, MIT.
The rsonpath crate provides a JSONPath parser and a query execution engine, which utilizes SIMD instructions to provide massive throughput improvements over conventional engines. It is the software behind the ASPLOS paper [20] which includes benchmarks. The project is under very active development with a strong potential to reach a large audience considering the topic.

- **Software Family**: Software as a Vehicle for Research.
- Audience: `partners`
- Evolution and maintenance: `lts`.
- Duration of the Development (Duration): 2 years.

6 Transfer and Impact

The work of D-DAL has a strong theoretical component. Our main impact will consist in publications in our research communities, but also in neighbouring communities where our research can find applications. Apart from this standard dissemination of knowledge, we plan to develop software implementations that can be readily used in research and possibly transferred to industrial partners. We see implementations as a form of outreach, allowing us to get in touch with communities beyond our immediate neighbors. An example of such a transfer is the use of `networkdisk` in the Bonsai team: they use this tool to study biological sequences, with a PhD thesis that has recently started.

6.1 Transfer and impact in other scientific communities

Artificial Intelligence D-DAL's research addresses several issues related to general challenges in AI. D-DAL provides high quality data and gives tools to explain how data has been produced. Indeed, the quality of training and of test data is crucial in the learning process. D-DAL offers tools to control it, because our research takes into account data quality, like uncertainty or probabilities.

Finally, a major challenge with neural networks is their black-box nature. D-DAL aims to explore inference methods for schemas and queries based on neural networks, while offering models that are more interpretable, such as finite-state models. We will also study topics related to the explainability of query results, e.g., via notions such as Shapley values or data provenance, which can help address the challenge of AI explainability in a broader sense.

Data analysis D-DAL will develop tools that exploit the parallel capabilities of CPUs to develop efficient streaming algorithms for data. Together with composition and transducers methods, D-DAL's research will produce flexible tools which can define efficient data processing pipelines. This will bring the possibility to perform data analysis on smaller computers and thus allow for cheaper and more accessible data analytics.

Foudations of computer science D-DAL's research is strongly grounded in foundations of computer science: algebra, logic, automata theory, semantics, combinatorics, etc. We shall develop theoretical tools that are not limited to data processing and that may be applied to other areas of computer science. We are in particular thinking of areas such as verification, information theory, complexity theory, etc.

6.2 Direct scientific transfer

Biological sequences We are starting a collaboration with the team Bonsai in CRIStAL (the joint Computer Science laboratory of CNRS and Université de Lille). The collaboration focuses on the use of graph database methods to handle larges sets of sequences. We hope that these methods can be used to tackle problems in bioinformatics and in medicine. This collaboration is now formalized within an ANR project and a PhD student is working on this project. It involves the manipulation of large graphs of genetic data. We use and develop `NetworkDisk` to handle these graphs.

Transportation data We also plan to study bicycle usage at the level of the Métropole Européenne de Lille (MEL) to help data-driven urbanistic decisions concerning the deployment of secure bike lanes. We will integrate GPS traces and data from traffic sensors in order to extract information relevant for the analysis of the bike traffic in MEL. Preliminary discussions with MEL already allowed us to identify a number of problems of interest. We are currently constructing a prototype as a proof of concept. The next step will be to formalize this collaboration with MEL. The scientific questions are related to combining privacy preserving data integration and efficient evaluation of graph algorithms and queries on possibly probabilistic models. We believe that that our expertise on data will allow us to construct a fruitful partnership.

RDF schemas Our work on ShEx has a long story in our team. ShEx was a candidate for a W3C recommendation for a schema language for RDF data. In the end, the language SHACL has been chosen instead. However, ShEx is under standardization at the IEEE and is used at Wikidata. The development of ShEx is a joint effort with industrials, academia and international organizations. The main body of the standardization texts is writen by non academic partners of the working group. Our role in the standardisation process mainly consists in providing textual descriptions of the semantics of the language based on the scientific literature. With this work, the development of ShEx theory is being transfered to a large number of users via the standardization effort and also under the flagship of apache-jena for the ShEx validator.

7 Positioning and Collaborations

7.1 Local collaborations

SISE. As a joint team with UMR CRIStAL, D-DAL will be in the SISE group (*systèmes informatiques sûrs et efficace*: safe and efficient computer systems) where we will have the opportunity to collaborate with 2XS team and the Sycomore team which will give us input on the hardware and the programming languages aspects of this line of research.

Magnet. As for the machine learning methods we plan to use for identifying data structures, we will collaborate with teams at the *centre INRIA de l'Université de Lille*. We have historical ties with the team Magnet which has a strong expertise in machine learning methods for graphs.

Bonsai. We have contacts with the Bonsai team which specializes in the algorithmic treatment of biological sequence. More specifically, their expertise lies in treating the output of new generation DNA sequencer which produce small segments of DNA from the original sequenced DNA. They propose methods to explore these segments to find properties of the original sequence. The number of sequenced DNA, and the number of segments is huge. D-DAL's work on graph databases can have an important impact on this line of work. A joint PhD thesis on this topic is starting soon.

CRIL. We have contacts with the CRIL a computer science laboratory in Lens which, among other topics, specializes in knowledge compilation. We are going to strengthen our activities with them, via common projects, co-supervision of PhD students, ... Our goal is to construct a cross-fertilizing interaction between D-DAL and CRIL that will allow both of us to enhance our research perspectives and impact.

University of Lille The research at the University of Lille is structured around four research hubs. As a research team of the University of Lille, D-DAL contributes to the hub *Monde numérique au service de l'humain*. Indeed, our work focuses on developing the algorithmic foundations needed to organize and interpret the ever-growing mass of data.

7.2 Positioning within the INRIA eco-system

BOREAL. BOREAL's research is complementary from that of D-DAL. The overall aim to make value out of data is common to our project. In particular, we share the same attention to consider the variety of data. However, BOREAL is focused on higher level questions. In particular, BOREAL has a strong focus on reasoning with ontological knowledge. BOREAL studies powerful logics, notably rule based languages. We share some common methods coming from knowledge representations, but our focus on circuits and our directions on the resolutions of queries are different. Concerning data descriptions, we are at the level of schema constraints which are much simpler than the logics BOREAL uses to describe data properties. Our shared interests pushes us to have collaborations: we have had some in the past and are also working on new ones.

CEDAR. CEDAR's research is at the level of cloud data stores and specializes on the scalability of parallel big data storage and processing. D-DAL focuses more on sequential algorithms and tries to exploit local parallelism rather than clusters of machines. Moreover, D-DAL's work on querying relies on exact specifications of data to be extracted while CEDAR's develops user/data interactions. A connection between the two teams the interest for graphs of heterogeneous data, RDF data and ontology mediated query answering.

TYREX. TYREX is working in two main directions: the development of expressive and efficient query languages and neuro-symbolic programming with graphs. Its first main direction constitutes a shared research interest with D-DAL. We also share some fundamental tools like logic and programming language theory. D-DAL has a stronger focus on query algorithms. Furthermore, our axis based on circuits and their use in query resolution and in the compilation of queries is very different from the techniques used in TYREX. Our focus on functional and synchronous programming languages and transducers for streaming is also a difference in focus of D-DAL with TYREX.

VALDA. D-DAL shares with VALDA its main challenges and a focus on probabilistic, uncertain and dynamic data. We also share many formal tools when it comes to query answering. We have had contacts and collaboration with VALDA regularly in the past and will certainly have in the future. However, beyond query answering, D-DAL is willing to develop streaming methods for data that integrate fine-grained tuning by means of circuits and will also propose compositional methods to integrate streaming algorithms into efficient data processing pipelines.

WIMMICS. WIMMICS is interested in graph databases and web semantics. Its research directions meet D-DAL's scientific interests at the level of rdf graphs. D-DAL's research is more basic and algorithmic than that of WIMMICS. Our work on finding structures within data graphs with hybrid machine learning methods may be of interest to WIMMICS as it may allow for more efficient graph processing, but also constitute basic knowledge on data from which semantics can be further inferred by the methods developed in WIMMICS.

7.3 International positioning

The team's research is connected to lines of work pursued by other research groups internationally; we already collaborate with many of these groups. We now describe the positioning of the team relative to these groups. One of the distinguishing features of D-DAL at the international level is the combination of deep theoretical work with a connection with implementations that try to take advantage of hardware. Moreover the team tries to cover all the key technical and theoretical problems in connection with data processing pipelines.

Santiago (Chile). Santiago houses the Millennium Institute "Foundational Research on Data", which spans the Pontifical Catholic University of Chile and the University of Chile. The institute leads research efforts in database theory, in particular on graph databases (including recursive queries), string algorithms and declarative information extraction, counting algorithms, and connections to graph neural networks. Our team has established research collaborations with some of these researchers. In particular, Mikaël Monet has worked with Pablo Barcelo, Marcelo Arenas and others on the complexity of counting problems over incomplete data, Shapley values, and expressiveness questions in machine learning and AI. Antoine Amarilli has worked with Cristian Riveros and Martín Muñoz and others about declarative information extraction and questions related to enumeration for formal languages.

Warsaw and Wrocław (Poland). Charles Paperman has long-standing collaborations with researchers in Warsaw, Poland, such as Filip Murlak and Mateusz Gienieczko on the efficient vectorized evaluation of queries over JSON documents, and Filip Mazowiecki and Michal Pilipczuk on recursive sequences. The Institute of Informatics at Warsaw is central for the research done by the team, and we intend to further develop collaborations with them, for instance with Mikołaj Bojańczyk (logics, queries on structured data), Marcin Pilipczuk (graph theory), Szymon Toruńczyk (graph theory, query evaluation), Michał Skrzypczak (logic and trees), and Wojciech Czerwiński (automata theory).

Important research related to the team's objectives also takes place in Wrocław, with whom we intend to develop collaborations: in particular Paweł Gawrychowski and Artur Jez (formal languages and string algorithms), Witold Charatonik and Emanuel Kieronski (logical reasoning and finite model theory), and Tomasz Gogacz and Jerzy Marcinkowski (finite model theory and open-world query answering).

United States. Charles Paperman has a long-standing collaboration with Michaël Cadilhac (DePaul university) and has started working with Howard Straubing (Boston College) on formal language theory topics. Following a visit at the Simons Institute, Antoine Amarilli and Mikaël Monet have started a collaboration with Wolfgang Gatterbauer and Neha Makhija (Northeastern University) on questions related to query resilience, i.e., understanding how robust are query results under changes to the input data; and Antoine Amarilli has started a collaboration with Nicole Wein (University of Michigan) on similar topics. Antoine Amarilli and Mikaël Monet also collaborate with Dan Suciu (University of Washington) on probabilistic data management topics.

Haifa and Tel Aviv (Israel). Antoine Amarilli and Mikaël Monet have collaborations with Benny Kimelfeld (Technion) on query evaluation over probabilistic databases and the computation of Shapley values. Antoine Amarilli also has previous collaborations with Yael Amsterdamer (Bar-Ilan university), Daniel Deutch (Tel Aviv university), and Tova Milo (Tel Aviv university).

Oxford (United Kingdom). Antoine Amarilli has a long-standing collaboration with Michael Benedikt (University of Oxford) on questions related to open-world query answering and logic. He also has collaborated with Ismail Ilkan Ceylan (University of Oxford) about query evaluation over probabilistic databases.

Zürich (Switzerland). An important group for the team's research objectives on query evaluation and incremental maintenance is the Data Systems and Theory groups in Universität Zürich, headed by Dan Olteanu. We hope to develop collaborations with the group, in particular with Dan Olteanu and Ahmet Kara.

Oviedo (Spain) Iovka Boneva has a collaboration with Jose Labra Gayo from the University of Oviedo (Spain) on a the development of the ShEx language and related tooling.

8 Notable publications

Stackless Processing of Streamed Trees, C. Barloy, F. Murlak, C. Paperman, *PODS 2021 - Symposium on Principles of Database Systems* [11].

Processing tree-structured data in the streaming model is a challenge: capturing regular properties of streamed trees by means of a stack is costly in memory, but falling back to finite-state automata drastically limits the computational power. We propose an intermediate stackless model based on register automata equipped with a single counter, used to maintain the current depth in the tree. We explore the power of this model to validate and query streamed trees. Our main result is an effective characterization of regular path queries (RPQs) that can be evaluated stacklessly-with and without registers. In particular, we confirm the conjectured characterization of tree languages defined by DTDs that are recognizable without registers, by Segoufin and Vianu (2002), in the special case of tree languages defined by means of an RPQ.

Static Analysis of Graph Database Transformations I. Boneva, B. Groz, J. Hidders, F. Murlak, S. Staworko, *Symposium on Principles of Database Systems (PODS 2023)* [12].

We investigate graph transformations, defined using Datalog-like rules based on acyclic conjunctive two-way regular path queries (acyclic C2RPQs), and we study two fundamental static analysis problems: type checking and equivalence of transformations in the presence of graph schemas. Additionally, we investigate the problem of target schema elicitation, which aims to construct a schema that closely captures all outputs of a transformation over graphs conforming to the input schema. We show all these problems are in EXPTIME by reducing them to C2RPQ containment modulo schema; we also provide matching lower bounds. We use cycle reversing to

reduce query containment to the problem of unrestricted (finite or infinite) satisfiability of C2RPQs modulo a theory expressed in a description logic.

On the Complexity of SHAP-Score-Based Explanations: Tractability via Knowledge Compilation and Non-Approximability Results, M. Arenas, P. Barceló, L. Bertossi, M. Monet, *Journal of Machine Learning Research 24, 63, 2023, p. 1–58* [10].

Scores based on Shapley values are widely used for providing explanations to classification results over machine learning models. A prime example of this is the influential SHAPscore, a version of the Shapley value that can help explain the result of a learned model on a specific entity by assigning a score to every feature. While in general computing Shapley values is a computationally intractable problem, we prove a strong positive result stating that the SHAP-score can be computed in polynomial time over deterministic and decomposable Boolean circuits under the so-called product distributions on entities. Such circuits are studied in the field of Knowledge Compilation and generalize a wide range of Boolean circuits and binary decision diagrams classes, including binary decision trees, Ordered Binary Decision Diagrams (OBDDs) and Free Binary Decision Diagrams (FBDDs). Our positive result extends even beyond binary classifiers, as it continues to hold if each feature is associated with a finite domain of possible values.

Supporting Descendants in SIMD-Accelerated JSONPath, M. Gienieczko, F. Murlak, C. Paperman, *International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2024)* [20].

Harnessing the power of SIMD can bring tremendous performance gains in data processing. In querying streamed JSON data, the state of the art leverages SIMD to fast forward significant portions of the document. However, it does not provide support for descendant, which excludes many real-life queries and makes formulating many others hard. In this work, we aim to change this: we consider the fragment of JSONPath that supports child, descendant, wildcard, and labels. We propose a modular approach based on novel depth-stack automata that process a stream of events produced by a state-driven classifier, allowing fast forwarding parts of the input document irrelevant at the current stage of the computation. We implement our solution in Rust and compare it with the state of the art, confirming that our approach allows supporting descendants without sacrificing performance, and that reformulating natural queries using descendants brings impressive performance gains in many cases.

Dynamic Membership for Regular Languages, A. Amarilli, L. Jachiet, C. Paperman, *International Colloquium on Automata, Languages and Programming* [6]. Best paper award of ICALP'21 (track B).

This work studies the question of query evaluation over dynamic data. In this work, the data is very simple: it is just a textual document, on which we can perform substitution updates to change individual characters. The language of queries is also very simple: we allow Boolean queries (returning a yes/no answer), which are defined by a regular language. In other words, the problem is the following: fixing a regular language, and given as input a word on which we can perform character substitutions, maintain after each substitution the information of whether the current word belongs to the language or not.

The originality of this work is to aim for a precise characterization of the complexity of this problem, defined to be the time required after each update, and expressed as a function of the word length. The complexity is generally logarithmic in the word length, but can be as low as constant, depending on the language. Our work identifies three complexity regimes for languages (constant-time, doubly logarithmic time, and singly logarithmic time): the classification criterion depends on involved notions of algebraic automata theory, and achieving the right upper bounds also require the use of efficient data structures from the algorithms literature.

9 Bibliography

References

[1] Antoine Amarilli, Marcelo Arenas, YooJung Choi, Mikaël Monet, Guy Van den Broeck, and Benjie Wang. A circus of circuits: Connections between decision diagrams, circuits, and automata. *arXiv preprint arXiv:2404.09674*, 2024.

[2] Antoine Amarilli, Pierre Bourhis, Florent Capelli, and Mikaël Monet. Ranked enumeration for MSO on trees via knowledge compilation. In *ICDT*, volume 290, 2024.

[3] Antoine Amarilli, Pierre Bourhis, Stefan Mengel, and Matthias Niewerth. Enumeration on trees with tractable combined complexity and efficient updates. In *PODS*, 2019.

[4] Antoine Amarilli and Florent Capelli. Tractable circuits in database theory. *ACM SIGMOD Record*, 53(2), 2024.

[5] Antoine Amarilli and İsmail İlkan Ceylan. The dichotomy of evaluating homomorphism-closed queries on probabilistic graphs. *LMCS*, 2022.

[6] Antoine Amarilli, Louis Jachiet, and Charles Paperman. Dynamic membership for regular languages. In *ICALP*, 2021.

[7] Antoine Amarilli and Mikaël Monet. Weighted counting of matchings in unbounded-treewidth graph families. *arXiv preprint arXiv:2205.00851*, 2022.

[8] Antoine Amarilli, Mikaël Monet, and Dan Suciu. The non-cancelling intersections conjecture. *CoRR*, abs/2401.16210, 2024.

[9] Antoine Amarilli and Mikaël Monet. Enumerating regular languages with bounded delay. In *STACS*, 2023.

[10] Marcelo Arenas, Pablo Barceló, Leopoldo Bertossi, and Mikaël Monet. On the complexity of SHAP-score-based explanations: Tractability via knowledge compilation and non-approximability results. *JMLR*, 24(63), 2023.

[11] Corentin Barloy, Filip Murlak, and Charles Paperman. Stackless processing of streamed trees. In *PODS*, 2021.

[12] Iovka Boneva, Benoit Groz, Jan Hidders, Filip Murlak, and Slawomir Staworko. Static analysis of graph database transformations. In *PODS*, 2023.

[13] Jin-Yi Cai, Pinyan Lu, and Mingji Xia. Holant problems and counting CSP. In *STOC*, 2009.

[14] Adnan Darwiche and Pierre Marquis. A knowledge compilation map. *Journal of Artificial Intelligence Research*, 17, 2002.

[15] Daniel Deutch, Nave Frost, Benny Kimelfeld, and Mikaël Monet. Computing the Shapley value of facts in query answering. In *SIGMOD Conference*. ACM, 2022.

[16] Martin Dyer and David Richerby. An effective dichotomy for the counting constraint satisfaction problem. *SIAM Journal on Computing*, 42(3), 2013.

[17] Ronald Fagin, Benny Kimelfeld, Frederick Reiss, and Stijn Vansummeren. Document spanners: A formal approach to information extraction. *JACM*, 62(2), 2015.

[18] Cibele Freire, Wolfgang Gatterbauer, Neil Immerman, and Alexandra Meliou. The complexity of resilience and responsibility for self-join-free conjunctive queries. *PVLDB*, 9(3), 2015.

[19] Lily Gallois. *Dialog between chase approach and string rewriting system approach.* Theses, Université de Lille, December 2019.

[20] Mateusz Gienieczko, Filip Murlak, and Charles Paperman. Supporting Descendants in SIMD-Accelerated JSONPath. In *ASPLOS*, 2024.

[21] Kathrin Hanauer, Monika Henzinger, and Christian Schulz. Recent advances in fully dynamic graph algorithms: A quick reference guide. *JEA*, 27, 2022.

[22] Mark Jerrum. Counting constraint satisfaction problems. In *The Constraint Satisfaction Problem*, volume 7 of *Dagstuhl Follow-Ups*. 2017.

[23] Pratik Karmakar, Mikaël Monet, Pierre Senellart, and Stéphane Bressan. Expected Shapley-like scores of boolean functions: Complexity and applications to probabilistic databases. *PACMMOD*, 2(2), 2024.

[24] Wim Martens and Tina Trautner. Evaluation and enumeration problems for regular path queries. In *ICDT*, 2018.

[25] Charles Paperman, Sylvain Salvati, and Claire Soyez-Martin. An algebraic approach to vectorial programs. In *STACS*, 2023.

[26] Sylvain Salvati and Sophie Tison. Containment of Regular Path Queries Under Path Constraints. In *ICDT*, 2024.

[27] Nicole Schweikardt, Luc Segoufin, and Alexandre Vigny. Enumeration for FO queries over nowhere dense graphs. *JACM*, 69(3), 2022.

[28] Luc Segoufin. Constant delay enumeration for conjunctive queries. *ACM SIGMOD Record*, 44(1), 2015.

[29] Pierre Senellart, Louis Jachiet, Silviu Maniu, and Yann Ramusat. ProvSQL: Provenance and probability management in PostgreSQL. *PVLDB*, 11(12), 2018.

[30] Claire Soyez-Martin. *From semigroup theory to vectorization: recognizing regular languages*. Theses, Université de Lille, December 2023.

10 Appendix

Dr. Antoine Amarilli, Advanced Research Position at INRIA Lille

Education

Qualification professeur des universités (CNU section 27), 2024

HDR, Institut Polytechnique de Paris, Paris, France. Computer science, 2023

PhD, Télécom Paris, Paris, France. Computer science, 2016

MS MPRI, École normale supérieure, Paris, France, 2012

Admission to École normale supérieure, Paris, France, 2009

Professional experience

2024-09-01–...: Advanced Research Position, Inria Lille, team LINKS

2016-08-01–2024-08-31: Associate professor, Telécom Paris

Professional activities

Main pedagogical responsibilities. Local representative of Télécom Paris in the MPRI Master of Science; oral examiner in computer science in École normale supérieure (2017–2021) and École polytechnique (2023–2024); director of the ICPC SWERC programming contest (2017–2021, 228-321 participating students)

Main program committee memberships ICDT'25, ICDT'24, ICALP'22, KR'21 , AAAI'21, IJCAI'21, STACS'21, IJCAI'20, IJCAI'19, ICDT'19, ICDT'18

Awards

- 2022 Ranked 321st worldwide at Round 3 of the Google Code Jam programming contest.
- 2021: Best paper award at ICALP'21, track B (with L. Jachiet and C. Paperman).
- 2020: Best paper award at ICDT (with I. Ilkan Ceylan).
- 2017: Joint winner of the Beth Dissertation Award.
- 2017: Placed first at the Télécom Paris PhD prize.
- 2015: Placed first at the Google Hash Code programming contest (by teams)

Recent Projects Member of the ANR projects: CQFD (2018–2024), EQUUS (2019–2025).

Selected publications

- Antoine Amarilli, Timothy van Bremen, Kuldeep S. Meel. Conjunctive Queries on Probabilistic Graphs: The Limits of Approximability. ICDT'24.
- Antoine Amarilli, Mikaël Monet. Enumerating Regular Languages with Bounded Delay. STACS'23.
- Antoine Amarilli, Louis Jachiet, Charles Paperman. Dynamic Membership for Regular Languages. Best paper award of ICALP'21 track B.
- Antoine Amarilli, İsmail İlkan Ceylan. A Dichotomy for Homomorphism-Closed Queries on Probabilistic Graphs. Best paper award of ICDT'20.
- Antoine Amarilli, Pierre Bourhis, Stefan Mengel, Matthias Niewerth. Constant-Delay Enumeration for Nondeterministic Document Spanners. ICDT'19 Featured in ACM SIGMOD Research Highlights.

Dr. Iovka Boneva, Associate professor at University of Lille

Education

PhD Université des Sciences et Technologies de Lille, Computer Science, 2006

Master (DEA) in Computer Science, Université des Sciences et Technologies de Lille, 2002

Professional experience

2008– : Associate professor in Computer Science, Université de Lille, Institute of Technology

2006–2008 : Post-doc researcher at Universiteit Twente, The Netherlands

2005–2006 : Research and teaching assistant (ATER), Université des Sciences et Technologies de Lille

Professional activities

Teaching Assumes a teaching duty of 200–230 hours/year to bachelor students at the Institute of Technology of University of Lille. Teaches mainly programming and algorithmics, software development and graph algorithms. Supervises projects and internships.

Research Co-supervisor of the PhDs of Momar Sakho (2016–2020) and Jose Martin Lozano Aparicio (2016–2020). Member of the W3C Data shapes working group (2015–2018), W3C Shape Expressions community group (2018–). Member of the program committees of ADBIS (2013, 2014), AMW (2017, 2018, 2019, 2024), EDBT Vision Track (2016), ESWC Resources Track (2023), SPIRE (2021).

Administration Has been member of the following councils and commissions: Research lab council (local); Inria Lille committee (local); Animator of the CRIStAL research lab committee for sustainable development (local); approx. 10 recruitment commissions for Associate professor or Research associate positions (local, other French universities); steering committee of the BDA, the French conference on management of Data (national).

Recent Projects

ANR Datacert 2015–2020

Selected publications

- Iovka Boneva, Benoît Groz, Jan Hidders, Filip Murlak, Slawek Staworko. Static Analysis of Graph Database Transformations. PODS 2023.
- Iovka Boneva, Joachim Niehren, Momar Sakho. Regular matching and inclusion on compressed tree patterns with constrained context variables. Inf. Comput. 286: 104776 (2022)
- Iovka Boneva, Slawek Staworko, Jose Lozano. Consistency and Certain Answers in Relational to RDF Data Exchange with Shape Constraints. ADBIS 2020: 97-107
- José Emilio Labra Gayo, Eric Prud'hommeaux, Iovka Boneva, Dimitris Kontokostas. Validating RDF Data. Synthesis Lectures on the Semantic Web: Theory and Technology, Morgan & Claypool Publishers 2017, ISBN 978-3-031-79477-3
- Iovka Boneva, José Emilio Labra Gayo, Eric G. Prud'hommeaux. Semantics and Validation of Shapes Schemas for RDF. ISWC (1) 2017: 104-120.

Dr. Aurélien Lemay, Assistant Professor at University of Lille

Education

HDR Université de Lille, France, Computer Science, 2018

PhD Université de Lille, France, Computer Science, 2002

MS (DEA) Université de Lille, France, Computer Science 1999

Civil Engineering Institut Supérieur d'Electronique du Nord, Lille, 1999

Professional experience

2003– : Associate professor in Computer Science, Université de Lille

2002–2003 : Research and teaching assistant (ATER), Université des Sciences et Technologies de Lille

Professional activities

Teaching Assumes a teaching duty of 200–230 hours/year to bachelor students at the Applied Foreign Languages department of the University of Lille. Teaches mainly databases, website design, programming fundamentals, and office automation tools. Coordinates teaching activities in computer science within the department since 2004.

Research Co-supervisor of the PhDs of Paul Gallot (2021), Adrien Boiret (2016), Radu Ciucianu (2015), Antoine Mbaye-Ndione (2015), Grégoire Laurence (2014), and Jérôme Champavère (2010). Participated in the ANR COLIS project (2015–2021). Member of the program committees of ICGI (2020, 2022, 2024).

Administration Coordination of computer science teaching in the Applied Foreign Languages department (local); Responsible of C2i and Pix in the same department; participation in the ANR COLIS project (national); several recruitment commissions for Associate professor or Research associate positions (local).

Selected publications

- Guillaume Bagan, Angela Bonifati, Radu Ciucanu, George H. L. Fletcher, Aurélien Lemay, Nicky Advokaat. gMark: Schema-Driven Generation of Graphs and Queries. IEEE Trans. Knowl. Data Eng. 29(4): 856-869 (2017).
- Joachim Niehren, Jérôme Champavère, Rémi Gilleron, Aurélien Lemay. Query Induction with Schema-Guided Pruning Strategies. Journal of Machine Learning Research (JMLR), 14:927–964 (2013).
- Antoine Mbaye Ndione, Aurélien Lemay, Joachim Niehren. Approximate Membership for Regular Languages modulo the Edit Distance. Theoretical Computer Science (TCS), 487:37–49 (2013).
- Jérôme Champavère, Rémi Gilleron, Aurélien Lemay, Joachim Niehren. Efficient Inclusion Checking for Deterministic Tree Automata and XML Schemas. Information and Computation (IC), 207(11):1181–1208 (2009).
- Julien Carme, Rémi Gilleron, Aurélien Lemay, Joachim Niehren. Interactive Learning of Node Selecting Tree Transducers. Machine Learning (ML), 66(1):33–67 (2007). doi: 10.1007/s10994-006-9613-8.

Education

PhD, Télécom Paris, Paris, France. Computer science, 2018

Master MPRI, Université Paris 7, Paris, France, 2015

Ingénieur, École Nationale Supérieure des Mines de Nancy, Nancy, France, 2015

Professional experience

01/10/2020–: CRCN, Inria Lille, team LINKS

16/08/2023–15/12/2023: Research Fellow, Simons Institute (UC Berkeley campus)

01/01/2019–01/09/2020: Postdoctoral student, IMFD (Santiago, Chile)

Professional activities

I have been/am in the PC of the following conferences: AAAI 2021, ICDT 2022, PODS 2022, PODS 2025

Since January 2024 I am managing editor for the TheoretiCS journal, which is a peer-reviewed Diamond Open Access electronic journal covering all areas of computer science.

I regularly organize LINKs' team seminars

I regularly teach computer science at master level at University of Lille and Centrale Lille

I have been involved in the conception and/or correction of the 2022 and 2023 ENS/X entrance exams

Awards

- 2019: PhD thesis prize of the French databases community BDA (also awared to Michele Linardi).

Recent Projects

Selected publications

- Pratik Karmakar, Mikaël Monet, Pierre Senellart, Stéphane Bressan: Expected Shapley-Like Scores of Boolean functions: Complexity and Applications to Probabilistic Databases. PODS'2024
- Antoine Amarilli, Pierre Bourhis, Florent Capelli, Mikaël Monet: Ranked Enumeration for MSO on Trees via Knowledge Compilation. ICDT 2024
- Marcelo Arenas, Pablo Barceló, Leopoldo E. Bertossi, Mikaël Monet: On the Complexity of SHAP-Score-Based Explanations: Tractability via Knowledge Compilation and Non-Approximability Results. JMLR (2023)
- Antoine Amarilli, Mikaël Monet, Dan Suciu: The Non-Cancelling Intersections Conjecture. ArXiv preprint.
- Antoine Amarilli, Mikaël Monet: Enumerating Regular Languages with Bounded Delay. STACS 2023

Dr. Charles Paperman, Associate professor at Université de Lille

Education

HDR, Université de Lille, 2024.

PhD, Université Paris 7, 2014.

MS MPRI, Paris 7, Paris, France, 2011

Professional experience

Associate Professor at Lille university since 2017.

Délégation INRIA (2020–2021)

Professional activities

Awards

- 2021: Best paper award at ICALP'21, track B (with L. Jachiet and C. Paperman).

Recent Projects

- PI du Projet Région NetworkDisk
- PI Projet ANR Shannon meet Cray
- Contributor of ANR FindARN
- Contributor of ANR Kcoda

Selected publications

- *Supporting Descendants in SIMD-Accelerated JSONPath.* Mateusz Gienieczko, Filip Murlak, Charles Paperman. ASPLOS 2023
- *An Algebraic Approach to Vectorial Programs.* Charles Paperman, Sylvain Salvati, Claire Soyez-Martin. STACS 2023
- *The Regular Languages of First-Order Logic with One Alternation.* Corentin Barloy, Michaël Cadilhac, Charles Paperman, Thomas Zeume. LICS 2022
- *Dynamic Membership for Regular Languages.* Antoine Amarilli, Louis Jachiet, Charles Paperman. ICALP 2021
- *Stackless Processing of Streamed Trees.* Corentin Barloy, Filip Murlak, Charles Paperman PODS 2021

Dr. Sylvain Salvati, Full Professor in Computer Science at Université de Lille

Education

HDR Université de Bordeaux, France, Computer Science, 2015

PhD Institut National Polytechnique de Lorraine, France, Computer Science, 2005

MS (DEA) Université Henri Poincarré, Nancy, France, Computer Science 2001

Civil Engineering École Nationale Supérieure des Mines de Nancy, 2001

Professional experience

09/2016-... Full Professor in Computer Science at Université de Lille

02/2007-08/2016 CR at INRIA

06/2006-01/2007 Postdoc at the National Institute of Informatics, Tokyo Japan

11/2005-04/2006 Postdoc at the National Institute of Informatics, Tokyo Japan 2005-2006 Postdoc NII - Japon; 2005 Thèse.

Professional activities

PhD supervision Corentin Barloy (2021-2024), Claire Soyez-Martin (2020-2023), Paul Gallot (2017-2021), Jérôme Kirman (2011-2014), Pierre Bourreau (2007 2012).

Main pedagogical responsabilities Director of studies of the Master Informatique (2022-...) (250 étudiants), co-Director of studies of the master Master MIAGE (2017-2022) (100 étudiants), Director of studies of the second year of Licence Informatics and Mathematics (2020-2022) (15 étudiants).

Main administrative duties Elected Member of the Computer Science Department (2018-...), Elected Member of the Commission d'Évaluation INRIA (2019-...), Head of the team LINKS (2022-).

Recent Projects Member of the ANR projects: KCODA (2021-2025), Colis (2015-2021), Delta (2016-2021), Bravas(2017-2023).

Selected publications

- C. Paperman, S. Salvati et C. S. Martin. "An algebraic approach to vectorial programs". In STACS 23. 2023.
- K. Gebhardt, F. Meunier et S. Salvati. "On is an n-MCFL". In : J. Comput. Syst. Sci. 127 (2022), p. 41-52.
- J. Niehren, S. Salvati et R. Azimov. "Jumping Evaluation of Nested Regular Path Queries". In : Proceedings 38th International Conference on Logic Programming, ICLP 2022. Sous la dir. d'Y. Lierler, J. F. Morales, C. Dodaro, V. Dahl, M. Gebser et T. Tekle. T. 364. EPTCS. 2022, p. 79-92.
- P. Gallot, A. Lemay et S. Salvati. "Linear High-Order Deterministic Tree Transducers with Regular Look-Ahead". In : 45th International Symposium on Mathematical Foundations of Computer Science, MFCS 2020.
- S. Salvati et I. Walukiewicz. "Simply typed fixpoint calculus and collapsible pushdown automata". In : Math. Struct. Comput. Sci. 26.7 (2016), p. 1304-1350.

Dr. Sophie Tison, Professor Emeritus in Computer Science at Université de Lille

Education

HDR Université de Lille, France, Computer Science, 1991

PhD Université de Lille, France, Computer Science, 1983

MS Université Paris 7, France, Computer Science 1981

MS Bachelor, Master, Mathematics, Université Paris 6 1976-1978, "Agrégation" Mathematics 1979

École Nationale Supérieure Fontenay-aux-Roses, Mathematics and Computer Science 1976-1981

Professional experience

08/2022-... Professor Emeritus in Computer Science at Université de Lille

09/1991-07/2022 Full Professor in Computer Science at Université de Lille

09/1985-08/1991 Associate professor in Computer Science at Université de Lille

01/1982-08/1985 Assistant professor in Computer Science at Université de Lille

Professional activities

PhD supervision of 13 PhD students from 1989 to 2022

Main pedagogical responsabilities Strongly Involved in the Master MIAGE from 1986 to 1997 , in charge of the option "Computer Science" in the committee for "Agrégation de Mathématiques" from , member of the committee of CAPES "NSI" from

Main administrative duties Member of the management team of I-Site Université de Lille Nord-Europe in charge of Partnerships and Innovation 2017-2022, President of the digital innovation hub CITC 2016-2018, Vice President of Université Lille 1 for Partnerships and Innovation 2015-2016, Chair for the scientific and technical council of business and research cluster in retail domain named PICOM 2008-2014

Elected Member of the CNU (National Council of the Universités) for Computer Science 2019-2022 and of The National Committee for Computer and Information Science 2012-2016

Elected Board member of Université de Lille 2017-2021 and of Université Lille 2014-2015, elected scientific council member of Université Lille 1 2006-2010

Dean of the Research Department in Computer Science of University of Lille 2008-2014 (Deputy Dean 2001-2007), Director of PhD School "Engineering Sciences" of University of Lille 2000-2005.

Awards

2023 Awarded by HDF100 HDFID/French Tech Lille

2022 Knight of the national order of merit

2018 Grand prix Kuhlmann by the SSAAL

2017 Commander of the Order of Academic Palms

2010 Test Of Time Awards for "Dauchet, Sophie Tison: The Theory of Ground Rewrite Systems is Decidable" presented at LICS 1990

Recent Projects Member of the ANR projects: CQFD (2018–2024), COLIS (2015–2021).

Selected publications

- Sylvain Salvati, Sophie Tison Containment of Regular Path Queries Under Path Constraints 27th International Conference on Database Theory, ICDT'2024
- Olivier Bournez, Gilles Dowek, Rémi Gilleron, Serge Grigorieff, Jean-Yves Marion, Simon Perdrix, S. Tison Theoretical Computer Science: Computability, Decidability and Logic A Guided Tour of Artificial Intelligence Research - Volume III: Interfaces and Applications of Artificial Intelligence, pp.1-50, 2020
- Pierre Bourhis, Michel Leclère, Marie-Laure Mugnier, Sophie Tison, Federico Ulliana, Lily Gallois Oblivious and Semi-Oblivious Boundedness for Existential Rules IJCAI'2019, pp.1581-1587
- Meghyn Bienvenu, Pierre Bourhis, Marie-Laure Mugnier, Sophie Tison, Federico Ulliana Ontology-Mediated Query Answering for Key-Value Stores IJCAI'2017, pp 844-851