Asymptotic Determinacy of Path Queries using Union-of-Paths Views

Nadime Francis

Asymptotic Determinacy of Path Queries using Union-of-Paths Views

*Nadime Francis**
ENS-Cachan, Inria

Abstract

We consider the view-based query determinacy problem over graph databases for queries defined as unions of path queries. These queries select pairs of nodes in a graph that are connected through a path whose length falls in a given set. A view specification is a set of such queries. We say that a view specification $\mathbf{V}$ determines a query Q if, for all databases D, the answers to $\mathbf{V}$ on D contain enough information to answer Q. Our main result states that, given a view $\mathbf{V}$, there exists an explicit bound that depends on $\mathbf{V}$ such that we can decide the determinacy problem for all queries that ask for a path longer than this bound, and provide first-order rewritings for the queries that are determined. We call this notion asymptotic determinacy. As a corollary, we can also compute the set of almost all path queries that are determined by $\mathbf{V}$.

1 Introduction

View determinacy is a static analysis problem on databases that consists in deciding whether a given set of initial queries, called a view, contains enough information to answer a new query, and this on all databases. Solving this problem has many applications, namely in query optimization and caching. Assume that querying the database is costly, but that answers to all previous queries are kept in cache. Then it is useful to know whether a new query can be answered using only cached information and without accessing the database. In that case, we are then interested in finding a rewriting, that is, a query that explicitly computes the desired information by querying only cached data. Query determinacy can also be stated as a security problem. Assume that views represent information that can be publicly accessed, but that the considered query contains private data that should not be disclosed. Then it should be ensured that the view does not determine the query, in other words that there exists no rewriting of the private queries using public data.

We consider this question over graph databases. Graph databases are relational databases in which all relations are binary. Equivalently, they can be seen as directed graphs whose edges carry a label from a finite alphabet. Such databases arise naturally in several scenarios, which include social networks, crime detection, biological data and the Semantic Web. For instance, in social networks, individual data such as name or phone number are represented as nodes, whereas relationships between members of the network are edges linking the corresponding nodes and labeled by the nature of the relationship. Thus, a person X is a friend of a person Y if there is an edge going from X to Y with the label *friend*.

Information contained in a graph database does not only lie in the content of the graph but also in its topology, that is, in *how* the different data nodes are connected to each other. Typical queries then ask about topological properties of the graph, namely the existence of edges, paths, and so on. In a social network, a user X could be interested in computing the transitive closure of the *friend* relation: she would like to retrieve all nodes Y such that there is a path going from X to Y using only the *friend* label.

*Part of this work was done while the author was supported by EPSRC grant EP/M025268/1.

The determinacy problem has been considered in various contexts (see [1], [8] among others). It was shown in [2] that determinacy is decidable when queries and views are defined as path queries, that is, queries Q_k that select pairs of nodes (x, y) such that there is a path from x to y whose length is a given integer k. For instance, it proves that the two views Q_3 and Q_4 determine the query Q_5, which is not immediate to see.

The main contribution of our work is to extend this result by considering a broader class of views that allows disjunction. More precisely, we consider here unions of path queries. A union of path queries Q is a query that selects pairs of nodes in a graph that are connected through a path whose length falls in a given set. For instance, a query $Q_{k,\ell}$ selects pairs of nodes that are linked either by a path of length k or a path of length ℓ. A typical case covered by our work is the following: views are Q_2, $\mathrm{Q}_{1,2}$ and $\mathrm{Q}_{2,3}$, and we will see that these views determine the query Q_5. While most of our work can and should be understood in this setting, we also show how to extend it to cover arbitrary disjunctions, that is, unions of path queries whose associated sets are infinite.

In this paper, we show that, given a view $\mathbf{V}$ defined by unions of path queries, we can decide whether $\mathbf{V}$ determines a path query Q assuming Q is "big enough," that is, Q asks for the existence of a path longer than some n_0 that we can effectively compute from $\mathbf{V}$. We call this notion *asymptotic determinacy*. Although n_0 is of exponential size, our decision procedure actually works in Π_2^{P} in the size of the queries that occur in the view. When it concludes that $\mathbf{V}$ determines Q, we also provide a first-order rewriting of Q using $\mathbf{V}$. Otherwise, it produces a generic counter-example that shows that $\mathbf{V}$ does not determine Q. Our technique starts by reducing $\mathbf{V}$ to a much simpler view $\mathbf{V}'$ which has many useful properties, including that some $Q \in \mathbf{V}'$ is actually a path query Q_c, for some integer c. This particular query is a key to our reasoning, as it allows us to reduce infinite structures to finite ones by computing modulo c. The finite number of small queries that we are not able to process are cases where both our criterion of determinacy and our generic counter-examples fail.

Related Work.

The determinacy problem has been considered in [3] for regular path queries, i.e. queries that select pairs of nodes that are connected through a path whose sequence of labels satisfies some regular expression. In [3], determinacy is known as *losslessness under the exact views assumption*. However, it is still unknown whether this property is decidable and what a good rewriting language could be. Note that, on a one-letter alphabet, regular path queries are actually weaker than infinite unions of path queries, and that finite unions of path queries are exactly regular path queries whose associated set is finite.

The determinacy problem has been solved in two specific cases. First, [2] showed how to decide whether a path view determines a path query and provides first-order rewritings of the query using the view when it is the case. The work presented here can be seen as an extension of [2], where we consider more expressive views, by allowing (possibly infinite) disjunction. In [2], a simple criterion is given: the view determines the query if and only if the view image of a simple path satisfying the query is connected. We will see in Example 1 and Example 2 that this decision criterion no longer applies here, as the view images in these examples are connected, but the view does not determine the query.

Second, [5] proved that regular path queries can always be rewritten as Datalog queries using regular path views, assuming *monotone determinacy*. Monotone determinacy is a stronger form of determinacy that is known to be decidable in this setting. It requires both determinacy and the fact that rewritings of the query using the view are monotone. Our work can be seen as an attempt to lift this monotonicity assumption, while still retaining some of the expressive power of regular path queries and views, such as disjunction and transitive closure. Of course, without assuming monotonicity there is no hope of finding a rewriting in Datalog, since it can only express monotone queries.

On the negative side, it was proved in [6] that determinacy is undecidable for conjunctive queries and conjunctive views in the *unrestricted case*, that is, when the databases are allowed to be infinite. This work has later been expanded in [7] to cover the standard case where databases

are required to be finite, while maintaining the undecidability result. While conjunctive queries and views are not directly comparable to the languages that we consider, these results strongly hint at the fact that determinacy remains a difficult problem even in seemingly simple cases.

The simplest cases for which determinacy is undecidable and that contain path queries and union-of-paths views are provided by [5]. There, it is shown that determinacy is undecidable for regular path queries, using either context-free path views or conjunctive regular path views. Admittedly, both settings offer considerably more expressive power than the query and view languages that we consider here, such as multiple labels in the alphabet, disjunctions in the query, and conjunctions in the view. However, to the best of our knowledge, there are no intermediate settings for which determinacy has been shown to be undecidable.

This paper is an extended version of [4]. This extended version contains all the details of the proofs that were missing or merely sketched in the conference version. Additionally, Section 3 also covers an alternate and more intuitive statement of the asymptotic determinacy problem that was defined in [4], and Section 5 discusses several possible extensions of this work, and the challenges that come with them.

2 Preliminaries

Graph Databases.

A *binary schema* σ is a finite set of binary relational predicates. A *graph database* D over σ is a relational structure over a binary schema σ. Alternatively, it can also be seen as a directed edge-labeled graph whose labeling alphabet are symbols in σ. An element in the domain of D is called a *node*.

A *path* π in a database D from x_0 to x_m is a finite sequence $\pi = x_0a_0x_1 \dots x_{m-1}a_{m-1}x_m$, where each x_i is a node of D, each a_i is a symbol of σ, and for all i, $a_i(x_i, x_{i+1})$ holds in D. Such a path is said to be *simple* if each node occurs at most once in π. We write $\pi = x_0 \dots x_m$ when the specific symbol of σ that holds for each pair (x_i, x_{i+1}) is irrelevant. The length of π, denoted by $|\pi|$, is the length of the word $a_0 \dots a_{m-1}$, in this case m. To denote the fact that π is a path from x_0 to x_m, we will write $x_0 \xrightarrow{\pi} x_m$. Similarly, we will sometimes use $x \xrightarrow{k} y$ to say that there exists a path of length k going from x to y, when the sequence of symbols that occur along the path is irrelevant. By abuse of notation, we also consider π as a graph database that contains exactly the nodes $x_0, \dots, x_m$, and in which only $a_i(x_i, x_{i+1})$ holds, for all i.

Queries.

A binary *query* Q over a schema σ is a mapping associating to each graph database D over σ a binary relation Q(D) over the domain of D. We consider the following two query languages.

A *path query* Q is defined by a single positive integer k. On a given database D, Q returns all the pairs of nodes (x, y) in D such that there exists a path in D from x to y of length k. In other words, $\mathrm{Q(D)} = \{(x, y) \in \mathrm{D} \mid \exists \pi, x \xrightarrow{\pi} y \text{ and } |\pi| = k\}$. For ease of notation and consistency with what follows, we will write $\mathrm{Q} = \{k\}$.

A *union of path queries* Q is defined by a finite set of positive integers $\{k_1, k_2, \dots, k_n\}$, and returns all the pairs of nodes that are connected through a path whose length belongs to the set, i.e. $\mathrm{Q(D)} = \{(x, y) \in \mathrm{D} \mid \exists \pi, x \xrightarrow{\pi} y \text{ and } |\pi| \in \mathrm{Q}\}$. By abuse of notation, Q represents both the query and the associated set. Unions of path queries are a generalization of path queries, as any path query can be seen as a union of path queries whose associated set is of size 1.

Remark that these query languages cannot distinguish the specific labels that are used to build the required paths. The simplest way to compare them to other commonly used query languages is to assume that the schema σ on wich the databases are defined only contains a single predicate. In this case, path queries are conjunctive queries whose underlying graph is a directed path. Alternatively, they are also regular path queries whose associated language is a single word. Then, unions of path queries are simply unions of such queries, or regular path queries whose associated language is finite.

Fig. 1: A database (left) and its view image (right) through a view $\mathbf{V}$. Here, $\sigma = \{a\}$, $\tau = \{b, c\}$. $\mathbf{V} = \{\mathrm{Q}_b, \mathrm{Q}_c\}$, with $\mathrm{Q}_b = \{2\}$ and $\mathrm{Q}_c = \{3\}$. The names of the nodes are omitted for readability. Instead, the corresponding nodes are placed at the same relative position in the database and its view image.

If $|\sigma| \geq 2$, then path queries are *not* strictly speaking conjunctive queries. Indeed, assume that $\sigma = \{a, b\}$, then the query $\mathrm{Q} = \{2\}$ selects all pairs of nodes (x, y) such that there exists a path $\pi = x\alpha z\beta y$, with $\alpha\beta \in \{aa, bb, ab, ba\}$, which cannot be expressed as a conjunctive query. This means that path queries correspond to a very restricted form of disjunction. However, it is more natural to say that path queries are not directly evaluated on a graph database D, but on the underlying structure of D, which is an unlabeled graph. In this case, the first comparison applies.

Views.

Let σ and τ be two binary schemas. A *view* $\mathbf{V}$ from σ to τ is a set of binary queries over σ, one for each symbol in τ. Note that, since both σ and τ are finite, then $\mathbf{V}$ is also a finite set. Given a database D over σ, the *view image* of D, $\mathbf{V}(\mathrm{D})$, is defined as the database E over τ such that, for each $b \in \tau$, the interpretation of b in E is precisely $\mathrm{Q}_b(\mathrm{D})$, where Q_b is the query in $\mathbf{V}$ associated with b. The nodes of E are exactly those that appear in one of those relations, also known as the *active domain* of E. See Figure 1 for an example of a database and its view image.

By abuse of notation, we will use the same symbol (typically V) to denote both a symbol $b \in \tau$ and its associated query $\mathrm{Q}_b \in \mathbf{V}$. Thus, it is the same to say that $V(x, y)$ holds in $\mathbf{V}(\mathrm{D})$, or that $(x, y) \in V(\mathrm{D})$. In other words, the interpretation of (the symbol) V in $\mathrm{E} = \mathbf{V}(\mathrm{D})$, and the answers to (the query) V on D represent the same relation. Finally, for complexity purposes, we will use $|\mathbf{V}|$ to refer to the size of the set $\mathbf{V}$, that is, the number of queries it contains, and $\|\mathbf{V}\|$ to refer to the sum of sizes of all queries that occur in $\mathbf{V}$.

When a view is defined by queries that belong to a certain query language, we will informally extend the name of the query language to the view. For instance, a view defined by conjunctive queries will be called a conjunctive view, a view defined by path queries will be called a path view and a view defined by unions of path queries will be called a union-of-paths view.

Determinacy.

A formal definition of determinacy is given in [10] as:

Definition 1 (Determinacy)**.** *We say that a view $\boldsymbol{V}$ determines a query Q if:*

$$\forall D, D', \boldsymbol{V}(D) = \boldsymbol{V}(D') \;\Rightarrow\; Q(D) = Q(D')$$

Intuitively, this means that a view $\mathbf{V}$ determines a query Q, which we write $\mathbf{V} \twoheadrightarrow \mathrm{Q}$, if, for all databases D, $\mathbf{V}(\mathrm{D})$ always contains enough information to answer Q on D. Moreover, we say that a query R is a *rewriting* of Q using $\mathbf{V}$ if $\mathrm{R}(\mathbf{V}(\mathrm{D})) = \mathrm{Q}(\mathrm{D})$ for all D.

The *determinacy problem* is the problem of deciding, given a view $\mathbf{V}$ and a query Q, whether $\mathbf{V} \twoheadrightarrow \mathrm{Q}$. To the best of our knowledge, its decidability status is still open when Q is a conjunctive query and $\mathbf{V}$ a conjunctive view, and also when Q is a regular path query, and $\mathbf{V}$ a regular path view. Nonetheless, several cases have been considered and solved. The results in [2] solve the problem when Q is a path query and $\mathbf{V}$ a path view. In [10], this problem is considered for Q

and $\mathbf{V}$ defined by conjunctive queries, and in [5] for Q and $\mathbf{V}$ defined with regular path queries, both with the added restriction that $\mathbf{V}$ must determine Q in a monotone way, which means that a monotone rewriting of Q using $\mathbf{V}$ exists.

Here, we consider the determinacy problem for Q defined as a path query, and $\mathbf{V}$ defined as a set of unions of path queries. However, for each $\mathbf{V}$, there exist a finite number of queries Q on which our technique does not work. Hence, we are actually solving a slightly weaker problem, that we call the *α-asymptotic determinacy problem* which allows, for each $\mathbf{V}$, to exclude a finite number of queries Q. The excluded queries are those that ask for a path shorter than $\alpha(\mathbf{V})$, where α is a fixed function that maps each view to a natural number. Note that providing an answer or a rewriting in "almost all" cases is something that has already been considered in the same context, for instance in [2]. We can now formally state the problem and our main result:

PROBLEM :	α-ASYMPTOTIC DETERMINACY
INPUT :	A union-of-paths view $\mathbf{V}$ and a path query $\mathrm{Q} = \{n\}$, with $n > \alpha(\mathbf{V})$
QUESTION :	Does $\mathbf{V} \twoheadrightarrow \mathrm{Q}$?

Theorem 1. *There exists an explicit and computable function α for which the α-asymptotic determinacy problem is decidable. Moreover, when the view determines the query, the decision procedure effectively computes a first-order rewriting of the query using the view.*

It will actually come from the proof that the specific α for which we can solve the problem grows exponentially in the size of the queries in $\mathbf{V}$, as $\alpha(\mathbf{V})$ is in $2^{O(\|V\|^4)}$. However, the decision procedure itself works with a much lower Π_2^{P} complexity.

Arithmetic Tools.

Some of the proofs in this work involve a lot of arithmetic reasoning. We present here the notations and properties that we use. Given two integers n and d, $n[d]$ represents the remainder in the division of n by d. We say that two integers n_1 and n_2 are equivalent modulo d, and we write $n_1 \equiv n_2[d]$ if they have the same remainder modulo d. We denote by $\gcd(A)$ the greatest common divisor of a set of integers A, and we use $n_1 \wedge n_2$ for $\gcd(\{n_1, n_2\})$.

We will make extensive use of Bezout's Identity, a fundamental arithmetic property stating that, for a given set of integer $A = \{n_1, \ldots, n_k\}$, there exist integers $a_1, \ldots, a_k$ such that $a_1 n_1 + \ldots + a_k n_k = \gcd(A)$. Moreover, it was proved in [9] that these integers can be chosen so that $|a_i| \leq \max(|n_1|, \ldots, |n_k|)/2$, for all i. Remark also that if A contains both positive and negative numbers, then these integers can be chosen so that $a_i \geq 0$, for all i. We will refer to this as Bezout's Identity with positive coefficients. In this case, a naive way of producing such coefficients from those given by [9] makes them to be of size at most $|A| \max(|n_1|, \ldots, |n_k|)^2$, which will be sufficient for our analysis.

Additionally, for two binary relations R and S, we write $R \cdot S$ for $\{(x, z) \mid \exists y, R(x, y) \text{ and } S(y, z)\}$. Let n be a positive integer, we also use $R^1 = R$, and $R^n = R^{n-1} \cdot R$ if $n \geq 2$. For two unions of path queries $\mathrm{Q} = \{k_1, \ldots, k_n\}$ and $\mathrm{P} = \{\ell_1, \ldots, \ell_m\}$, we define $\mathrm{Q} \cdot \mathrm{P} = \{k_i + \ell_j \mid i \leq n, j \leq m\}$, and similarly define Q^n. This notation is consistent with the fact that, for any database D, $\mathrm{Q} \cdot \mathrm{P}(\mathrm{D}) = \mathrm{Q}(\mathrm{D}) \cdot \mathrm{P}(\mathrm{D})$. Remark that, when Q and P are path queries, then $\mathrm{Q} \cdot \mathrm{P}$ is also a path query, as is Q^n.

Organization.

The rest of the paper investigates the determinacy problem for a path query Q using view $\mathbf{V}$ defined by unions of path queries. In Section 3, we start by providing some conditions on Q and $\mathbf{V}$ that are necessary for $\mathbf{V}$ to determine Q, which leads to a more intuitive statement of the asymptotic determinacy problem. Section 4 is dedicated to proving Theorem 1, which gives a procedure for deciding the determinacy problem for almost all path queries Q, as well as a first-order rewriting of Q using $\mathbf{V}$ for the queries that are determined. Finally, in Section 5, we discuss the issue that remains to be solved in order to decide determinacy for all queries, as well as possible extensions of this work.

3 Towards Asymptotic Determinacy

In this section and the next, we consider path queries and union-of-paths views. When we simply say "query" or "view", it is implied that they belong to those specific classes.

In Section 3.1, we provide a set of necessary conditions for a view $\mathbf{V}$ to determine a query Q. We then use these results in Section 3.2 to restate the asymptotic determinacy problem in a more intuitive way, which will justify its name.

3.1 First Results

Our first lemma states that a view $\mathbf{V}$ cannot possibly determine a query Q if $\mathbf{V}$ does not at least contain a path query. In other words, even though $\mathbf{V}$ is defined using unions of path queries, at least one of them cannot make use of the union.

Lemma 1. *Assume that a view $\boldsymbol{V}$ and query Q are such that $\boldsymbol{V} \twoheadrightarrow Q$. Then there exists $C \in \boldsymbol{V}$ such that $|C| = 1$.*

Proof. Assume by contraposition that, for all $V \in \mathbf{V}$, $|V| > 1$. Let $\mathrm{Q} = \{n\}$. We build a database D over $\sigma = \{a\}$ as follows:

- D contains $n + 1$ distinct nodes $x_0, \ldots, x_n$.
- For all $i < n$, $a(x_i, x_{i+1})$ holds in D.
- For all $i \leq n$ and for all $V \in \mathbf{V}$ such that $i \in V$, we add to D a simple path $\pi_{i,V}$ from x_0 to x_i, such that $|\pi_{i,V}| \in V - \{i\}$. Such a path exists because $|V| > 1$.

We then construct another database D' which is a copy of D except that $a(x_0, x_1)$ does not hold in D'. It is then easy to check that $\mathbf{V}(\mathrm{D}) = \mathbf{V}(\mathrm{D}')$ and that $\mathrm{Q}(\mathrm{D}) \neq \mathrm{Q}(\mathrm{D}')$. In particular, $(x_0, x_n) \in \mathrm{Q}(\mathrm{D})$ and $(x_0, x_n) \notin \mathrm{Q}(\mathrm{D}')$. Hence $\mathbf{V} \not\twoheadrightarrow \mathrm{Q}$, which concludes the proof. This construction is illustrated in Figure 2. □

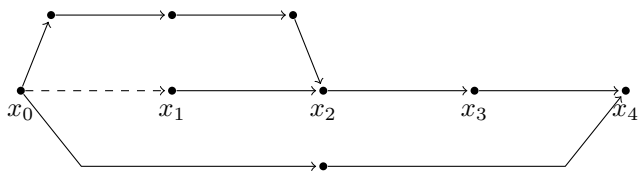

Fig. 2: Illustration for the proof of Lemma 1, showing here that $V = \{2, 4\} \not\twoheadrightarrow \mathrm{Q} = \{4\}$. Following the notations in the proof, the top path is $\pi_{2,V}$ and the bottom path is $\pi_{4,V}$. Remark then that adding or removing the dashed edge does not change the view, but changes the query result.

In view of Lemma 1, we know that we can restrict our attention to views that contain a path query C. This allows us to state the following definition for such views:

Definition 2 (Complete view)**.** *Let $\boldsymbol{V}$ be a view and $C \in \boldsymbol{V}$ such that $C = \{c\}$. We say that $\boldsymbol{V}$ is* C-complete *if, for all $i \in \{0, \ldots, c-1\}$, there exists $V \in \boldsymbol{V}$ and $k \in V$ such that $k \equiv i[c]$.*

For our next necessary condition, we adapt the condition in [2] which originally works as follows. Let $\mathrm{Q} = \{n\}$ be a path query and $\mathbf{V}$ be a path view, without unions. Consider a database D which consists of a simple path of length n: $x_0 \ldots x_n$. Let $\mathrm{E} = \mathbf{V}(\mathrm{D})$, and let G be the underlying undirected graph of E, that is a graph whose set of nodes is exactly the set of nodes of E, and such that there is an undirected and unlabeled edge between two nodes x and y in G if and only if there is any edge going from x to y or from y to x in E. Then it was shown in [2] that, if $\mathbf{V} \twoheadrightarrow \mathrm{Q}$, then x_0 and x_n belong to the same connected component in G. We remark that this condition still holds when $\mathbf{V}$ is a union-of-paths view:

Claim 1. *Let $\boldsymbol{V}$ be a view and $Q = \{n\}$. Let $\pi = x_0 \dots x_n$. If $\boldsymbol{V} \twoheadrightarrow Q$ then there is an undirected path from x_0 to x_n in $\boldsymbol{V}(\pi)$.*

This claim could quite easily be proved by looking at the construction given in [2] and showing that it still applies when $\mathbf{V}$ contains unions. Instead, we give another simple argument: consider $\mathbf{V}'$ to be the path view deduced from $\mathbf{V}$ by separating all the unions, that is, if $V = \{k_1, \dots, k_m\}$ is a query in $\mathbf{V}$, then $\mathbf{V}'$ contains all the queries $V_i = \{k_i\}$. Then the result comes from applying the condition on $\mathbf{V}'$ together with the two following remarks: first, if $\mathbf{V} \twoheadrightarrow \mathrm{Q}$, then $\mathbf{V}' \twoheadrightarrow \mathrm{Q}$, as $\mathbf{V}'$ yields strictly more information. Second, for any database D, $\mathbf{V}(\mathrm{D})$ and $\mathbf{V}'(\mathrm{D})$ have the same connectedness properties.

This condition allows us to reduce any determinacy problem to an equivalent problem with the added hypothesis that the view is C-complete:

Lemma 2. *Let $\boldsymbol{V}$ be a view and $C \in \boldsymbol{V}$ such that $C = \{c\}$. Let $Q = \{n\}$, and $\pi = x_0 \dots x_n$. Assume that there is a path from x_0 to x_n in $\boldsymbol{V}(\pi)$. Then we can effectively compute a view $\boldsymbol{V}'$ with $C' \in \boldsymbol{V}'$ such that $C' = \{c'\}$ and a query Q' such that $\boldsymbol{V}'$ is C'-complete and $\boldsymbol{V} \twoheadrightarrow Q$ if and only if $\boldsymbol{V}' \twoheadrightarrow Q'$.*

Proof. Let $\mathbf{V}$ and Q be defined as in the statement of the lemma. Let U be the set of all numbers that appear in $\mathbf{V}$, that is:

$$U = \bigcup_{\substack{V \in \mathbf{V} \\ u \in V}} \{u\}$$

and $d = \gcd(U)$.

Assume $d = 1$. Then there exists $u_1, \dots, u_k \in U$ such that $u_1 + \dots + u_k \equiv 1[c]$. This means that there exists $V_1, \dots, V_k \in \mathbf{V}$ and $v \in V = V_1 \cdot \dots \cdot V_k$ such that $v \equiv 1[c]$. We define $\mathbf{V}'$ as

$$\mathbf{V}' = \mathbf{V} \cup \bigcup_{i=1}^{c} \{V^i\}$$

and $\mathrm{Q}' = \mathrm{Q}$. It follows that $\mathbf{V}'$ is C-complete. Indeed, for all $k \in \{0, \dots, c-1\}$, V^k contains $kv \equiv k[c]$. Additionally, $\mathbf{V}' \twoheadrightarrow \mathrm{Q}'$ if and only if $\mathbf{V} \twoheadrightarrow \mathrm{Q}$, as all queries in $\mathbf{V}$ are also in $\mathbf{V}'$, and all queries in $\mathbf{V}'$ can be written as compositions of queries in $\mathbf{V}$.

Assume now that $d \neq 1$. Then d divides all the numbers in $\mathbf{V}$. Additionally, we know that there exists a path from x_0 to x_n in $\mathbf{V}(x_0 \dots x_n)$, which implies that d divides n as well. For each $V \in \mathbf{V}$, we define $V' = \{\frac{u}{d} \mid u \in V\}$. We then define $\mathbf{V}' = \{V' \mid V \in \mathbf{V}\}$ and $\mathrm{Q}' = \{\frac{n}{d}\}$.

Claim 2. *$\boldsymbol{V} \twoheadrightarrow Q$ if and only if $\boldsymbol{V}' \twoheadrightarrow Q'$.*

- Assume that $\mathbf{V} \not\twoheadrightarrow \mathrm{Q}$. Then there exists two databases D_1 and D_2 such that D_1 and D_2 agree on $\mathbf{V}$ but not on Q. We build two new databases D'_1 and D'_2 that are copies of D_1 and D_2 except that there is an edge between x and y in D'_i if and only if there is a path of length d from x to y in D_i.

  Let x and y be two nodes of D'_1 such that $(x, y) \in V'(\mathrm{D}'_1)$ for some $V' \in \mathbf{V}'$. Then there exists $u \in V'$ such that x and y are at distance u in D'_1. By construction, this means that x and y are at distance du in D_1. Hence, $(x, y) \in V(\mathrm{D}_1)$. Then $(x, y) \in V(\mathrm{D}_2)$. Thus, there exists $v \in V$ such that x and y are at distance v in D_2. It follows that x and y are at distance $\frac{v}{d}$ in D'_2, and finally that $(x, y) \in V'(\mathrm{D}'_2)$. Hence D'_1 and D'_2 agree on $\mathbf{V}'$. A similar reasoning shows that D'_1 and D'_2 don't agree on Q', so that we can conclude that $\mathbf{V}' \not\twoheadrightarrow \mathrm{Q}'$.

- Assume that $\mathbf{V}' \not\twoheadrightarrow \mathrm{Q}'$. Then there exists two databases D'_1 and D'_2 such that D'_1 and D'_2 agree on $\mathbf{V}'$ but not on Q'. We build two new databases D_1 and D_2 as follows:

  - For each node x of D'_i and each $\alpha \in \{0, \dots, d-1\}$, (x, α) is a node of D_i.
  - For each x and each $\alpha < d - 1$, there is an edge from (x, α) to $(x, \alpha + 1)$.

– For each x, y such that there is an edge from x to y in D'_i, there is an edge from $(x, d-1)$ to $(y, 0)$ in D_i.

Let (x, α) and (y, β) be two nodes of D_1 such that $\big((x, \alpha), (y, \beta)\big) \in V(\mathrm{D}_1)$ for some $V \in \mathbf{V}$. Then there exists $u \in V$ such that (x, α) and (y, β) are at distance u in D_1. Since $u \in V$, we have $u \equiv 0[d]$, which implies that $\alpha = \beta$. By construction, this implies that x and y are at distance $\frac{u}{d}$ in D'_1. Hence, $(x, y) \in V'(\mathrm{D}'_1)$. Then $(x, y) \in V'(\mathrm{D}'_2)$. Thus there exists $v \in V'$ such that x and y are at distance v in D'_2. By construction, this implies that (x, α) and (y, α) are at distance dv in D_2, and thus $\big((x, \alpha), (y, \alpha)\big) \in V(\mathrm{D}_2)$. Hence D_1 and D_2 agree on $\mathbf{V}$. A similar reasoning shows that D_1 and D_2 don't agree on Q, so that we can conclude that $\mathbf{V} \not\twoheadrightarrow \mathrm{Q}$.

Finally, we get a new set of views $\mathbf{V}'$ for which we can apply the first case of the proof and compute a new $\mathbf{V}'$ that is C'-complete. □

The last lemma of this section shows that the set of queries determined by a view $\mathbf{V}$ which contains a path query $C = \{c\}$ is closed under adding c. While perhaps obvious, this result is key to defining asymptotic determinacy, as will be explained in Section 3.2.

Lemma 3. *Let $\boldsymbol{V}$ be a view and $C \in \boldsymbol{V}$ such that $C = \{c\}$. Let $Q = \{n\}$, and assume that $\boldsymbol{V} \twoheadrightarrow Q$. Then, for all positive integers k, $\boldsymbol{V} \twoheadrightarrow \{n + kc\}$.*

Proof. Let D and D′ be two databases such that $\mathbf{V}(\mathrm{D}) = \mathbf{V}(\mathrm{D}')$. Let $\mathrm{Q}_1 = \{n+c\}$, and assume that there exists (x, y) in D, such that $(x, y) \in \mathrm{Q}_1(\mathrm{D})$. Then, there exists z in D such that $(x, z) \in \mathrm{Q}(\mathrm{D})$ and $(z, y) \in C(\mathrm{D})$.

Since $\mathbf{V} \twoheadrightarrow \mathrm{Q}$, then x appears in $\mathbf{V}(\mathrm{D})$, and thus in $\mathbf{V}(\mathrm{D}')$. Similarly, since $(z, y) \in C(\mathrm{D})$, then both z and y appear in both $\mathbf{V}(\mathrm{D})$ and $\mathbf{V}(\mathrm{D}')$. It remains to notice that $(x, y) \in \mathrm{Q}_1(\mathrm{D}')$. Indeed, since $\mathbf{V} \twoheadrightarrow \mathrm{Q}$, then $(x, z) \in \mathrm{Q}(\mathrm{D}')$, which implies that there exists a path of length n from x to z in D′. Additionally, $C(z, y)$ holds in $\mathbf{V}(\mathrm{D}')$ (because it also does in $\mathbf{V}(\mathrm{D})$), which implies that there exists a path of length c from z to y in D′. Altogether, these two paths combine to form a path of length $n + c$ going from x to y in D′, and conclude the proof that $\mathbf{V} \twoheadrightarrow \mathrm{Q}_1$.

It easily follows by induction that $\mathbf{V} \twoheadrightarrow \mathrm{Q}_k = \{n + kc\}$ for any positive integer k. □

3.2 Asymptotic Determinacy

In this section, we show how to use the results of Section 3.1 to assess the situation from a slighty different perspective. Let $\mathbf{V}$ be a view defined by unions of path queries. Assume that, instead of being given a specific query Q for which we want to decide whether $\mathbf{V} \twoheadrightarrow \mathrm{Q}$, we want to compute the complete determinacy picture of $\mathbf{V}$. In other words, we want to know all path queries Q such that $\mathbf{V} \twoheadrightarrow \mathrm{Q}$.

We start by using Lemma 1. This allows us to say that, if $\mathbf{V}$ does not contain a path query $C = \{c\}$, then there is no path query Q such that $\mathbf{V} \twoheadrightarrow \mathrm{Q}$, which answers our question, as well as the asymptotic determinacy problem. Let us now assume that $\mathbf{V}$ does indeed contain a path query $C = \{c\}$. Consider a natural number $o \in \{0, \ldots, c-1\}$. Then there are two cases for a query $\mathrm{Q} = \{m\}$ such that $m \equiv o[c]$:

- Case 1: $\mathbf{V}$ does not determine any query $\mathrm{Q}' = \{n\}$ with $n \equiv o[c]$. In particular, this means that $\mathbf{V} \not\twoheadrightarrow \mathrm{Q}$, which answers our question for such queries.
- Case 2: There exists some query $\mathrm{Q}' = \{n\}$ with $n \equiv o[c]$ such that $\mathbf{V} \twoheadrightarrow \mathrm{Q}$. Let us assume, without loss of generality, that Q′ is actually the smallest such query. Then either $m < n$, in which case we can easily conclude that $\mathbf{V} \not\twoheadrightarrow \mathrm{Q}$, or $m \geq n$, in which case Lemma 3 immediately proves that $\mathbf{V} \twoheadrightarrow \mathrm{Q}$. Thus the determinacy status for such queries is entirely determined by this specific n.

What this means is that, if we restrict our attention to *big enough* queries $\mathrm{Q} = \{m\}$ with $m \equiv o[c]$, there are only two possibilities. Either *none* of them are determined by Q (Case 1), or *all* of them are (Case 2). Thus, deciding the determinacy status of big enough queries becomes much easier: it simply amounts to deciding, for each $o \in \{0, \dots, c-1\}$, if it behaves as in Case 1 or Case 2. This is what we call the *asymptotic determinacy picture* of $\mathbf{V}$.

This gives a new perspective on the asymptotic determinacy problem: given a view $\mathbf{V}$, we can first compute the asymptotic determinacy picture of $\mathbf{V}$, and then compute a safety threshold $\alpha(\mathbf{V})$ that ensures that all queries Q that ask for paths longer than $\alpha(\mathbf{V})$ comply to this asymptotic determinacy picture. Then, given a query $\mathrm{Q} = \{n\}$ with $n > \alpha(\mathbf{V})$, it simply remains to check if $n[c]$ is in Case 1 or Case 2, which determines whether $\mathbf{V} \twoheadrightarrow \mathrm{Q}$. Finally, by using Lemma 2, we can restrict our attention to C-complete views. Altogether, this discussion shows that Theorem 1 is a consequence of the following proposition:

Proposition 1. *Given a C-complete view $\boldsymbol{V}$ defined by unions of path queries, such that $C \in \boldsymbol{V}$ with $C = \{c\}$ for some $c \in \mathbb{N}$ and a natural number $o \in \{0, \dots, c-1\}$, it is decidable in Π_2^P whether there exists a query $Q = \{n\}$ such that $n \equiv o[c]$ and $\boldsymbol{V} \twoheadrightarrow Q$. If this is the case, such a query Q of size $2^{O(\|\boldsymbol{V}\|^4)}$ and a first-order rewriting of Q with regards to $\boldsymbol{V}$ can be computed effectively.*

Indeed, if $\mathbf{V}$ is C-complete, the specific α required by Theorem 1 can be defined as the function that, given $\mathbf{V}$, computes a witness query for each $o \in \{0, \dots, c-1\}$ as in Proposition 1, and then returns the size of the biggest one. If $\mathbf{V}$ is not C-complete, then Lemma 2 allows us to compute a C-complete view $\mathbf{V}'$ for which the previous argument provides a suitable α'. It then remains to define $\alpha(\mathbf{V}) = k\alpha'(\mathbf{V}')$, where k is the gcd of all numbers that appear in $\mathbf{V}$, as in the proof of Lemma 2. Then, given a query $\mathrm{Q} = \{n\}$, with $n \geq \alpha(\mathbf{V})$, we define $\mathrm{Q}' = \{\frac{n}{k}\}$ if k divides n, and apply Proposition 1 with $\mathbf{V}'$ and Q', or conclude that $\mathbf{V} \not\twoheadrightarrow \mathrm{Q}$ otherwise.

Remark that, in order to produce the complete determinacy picture of $\mathbf{V}$, and thus to solve the general determinacy problem, we would need to compute the smallest query that is determined by $\mathbf{V}$ for each o. We do not know how to solve this challenging task yet, and discuss it further in Section 5.1. The proof of Proposition 1 itself is the goal of Section 4.

4 Deciding Asymptotic Determinacy

This section is devoted to proving Proposition 1, and therefore Theorem 1. For the whole section, we fix a C-complete view $\mathbf{V}$, with $C = \{c\} \in \mathbf{V}$ and a natural number $o \in \{0, \dots, c-1\}$, as in the statement of the proposition.

The proof is divided in three parts. In Section 4.1, we introduce a tool that describes the possible behaviors that can be observed through the view $\mathbf{V}$. We use it to prove the propositions in Section 4.2 and Section 4.3. More precisely, Section 4.2 settles the case where no query $\mathrm{Q} = \{n\}$ with $n \equiv o[c]$ is determined by $\mathbf{V}$, and Section 4.3 builds an appropriate query Q when one does exist.

4.1 Behavior Graph

The goal of this section is to define a tool that will help us deal with the combinatorial complexity of trying to find a path of specific length in a database based on its view image. We now give a rough sketch of the idea behind this tool. Assume we want to prove that some database D contains a path π of length n by looking only at $\mathrm{E} = \mathbf{V}(\mathrm{D})$. If D does indeed contain such a path, then the following properties must necessarily hold in E:

C1. E contains the $n+1$ (not necessarily distinct) nodes of π, $x_0, \dots, x_n$.

C2. For each $V \in \mathbf{V}$ and $u \in V$, $V(x_i, x_{i+u})$ holds in E for all i.

C3. For each x in E such that $V(x, x_i)$ holds in E, there exists an appropriate value of k and j such that $C^k(x, x_j)$ holds in E. The values of k and j depend on the witness path that proves $V(x, x_i)$, as shown in Figure 3.

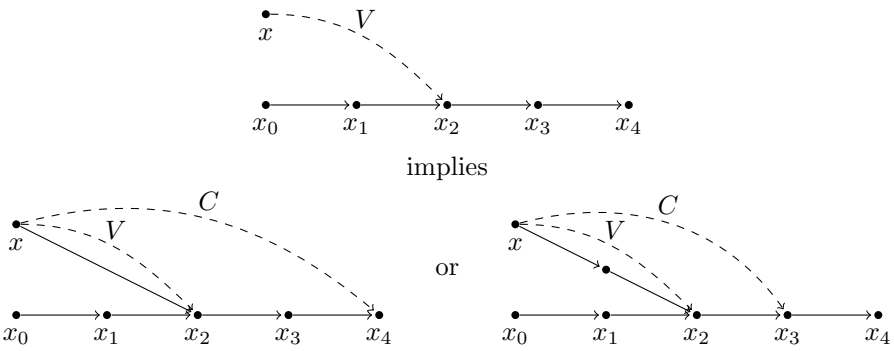

Fig. 3: Example of possible behaviors for a database (full) and its view (dashed), with $C = \{3\}$ and $V = \{1, 2\}$. Assume we know the information represented in the top figure. Then, one of the two bottom pictures must hold. More generally, if $C = \{c\}$ and $V(x, x_i)$ holds in E, then $C^k(x, x_j)$ must also hold, with $j = i + (c - v[c])$ and $kc = v + (c - v[c])$ for some $v \in V$.

If $\mathrm{E} = \mathbf{V}(\mathrm{D})$ does not satisfy these properties, then we can safely conclude that D does not contain a path of length n going from x_0 to x_n. Let us fix $x_0, \ldots, x_n$ and assume that E satisfies (C2) and (C3) for these fixed nodes. Our intention here is for these nodes to be the consecutive nodes of a path of length n going from x_0 to x_n. Unfortunately, there are still many ways for E to satisfy (C2) and (C3) without D actually having a path of length n from x_0 to x_n, let alone one that goes through all the x_i's in the right order.

Let μ be a path in D from some x_i to some x_j. We define the *delay* of this path as $\delta(\mu) = |\mu| - (j - i)$. The value $\delta(\mu)$ caracterizes the difference between μ and the section of the path of length n going through the x_i's, that we expected to find in D. If μ is of the intended $(j - i)$ length, then its delay will be zero. Otherwise $\delta(\mu)$ can be positive, if μ is longer than expected, or negative, if μ is shorter than expected.

Let D be a database and $\mathrm{E} = \mathbf{V}(\mathrm{D})$ such that E satisfies the necessary conditions above. For this D and E, we build a graph H_{D} that represents the delays of the paths of D that are induced by the conditions (C1), (C2) and (C3) as follows:

- H_{D} has $n + 1$ nodes that represent $x_0, \ldots, x_n$, as in (C1). We simply note them $0, \ldots, n$.
- For all $V \in \mathbf{V}$ and $u \in V$, (C2) implies that $V(x_i, x_{i+u})$ holds in E. Hence, there exists a path μ in D going from x_i to x_{i+u} of length v, for some $v \in V$. For each such μ:
  - We represent it as an edge in H_{D} going from i to $i + u$ of label $\delta(\mu) = (v - u)$.
  - For all $u' < v$ such that $u' \in V'$ for some $V' \in \mathbf{V}$, we know that $V'(x, x_{i+u})$ holds in E, where x is the node that occur at distance u' before x_{i+u} along μ. We apply (C3) as shown in Figure 4. This leads to a path μ' in D from x_i to $x_{i+u+(c-v'[c])}$ such that $\delta(\mu') = (v - u) + (v' - u')$, for some $v' \in V'$. We similarly represent each such μ' in H_{D}.

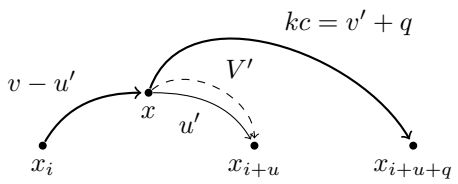

Fig. 4: Illustration for the existence of the path μ' of delay $(v - u) + (v' - u')$ in the construction of H_{D}. Full arrows represent paths in D and are labeled by their length. Dashed arrows represent edges in E. μ' is the thick path, and $q = c - v'[c]$.

Assume that there is a path from node 0 to node n in H_{D} whose sum of labels is 0. By composing all the paths in D that led to this path in H_{D}, we can prove that there exists in D a path π from x_0 to x_n such that $\delta(\pi) = 0$. Hence, π is of length n, and we have actually found a path of length n from x_0 to x_n in D.

Consider the case where this is true for all databases D, that is, for all databases D such that $\mathbf{V}(\mathrm{D})$ satisfies the necessary conditions, H_{D} contains such a path. Then all these databases contain a path of length n from x_0 to x_n. This means that the necessary conditions for the existence of a path of length n in D are also sufficient. Since these conditions can be checked by looking only at the view instance, it implies that $\mathbf{V} \twoheadrightarrow \{n\}$.

Unfortunately, the size of H_{D} depends on the size of the target query. In order to have a representation that does not depend on n, we identify in H_{D} all nodes i and j such that $i \equiv j[c]$. Note that this is consistent with the fact that such nodes were already linked by paths of delay 0 thanks to $C \in \mathbf{V}$. This is exactly the idea behind *choice graphs*, that are formally defined below. While we do lose some information by doing this merging, these graphs are still rich enough to allow us to decide asymptotic determinacy, as we will see in the rest of the proof.

Definition 3 (Choice graph)**.** *Given a C-complete view $\boldsymbol{V}$ such that $C \in \boldsymbol{V}$ with $C = \{c\}$, we define $\mathcal{H}_{\boldsymbol{V}}$ as the set of all directed, edge-labeled graphs H such that:*

1. *H has c nodes, which we will simply note $0, 1, \ldots, c-1$.*
2. *The edges of H carry labels in $\{-2(m-1), \ldots, 2(m-1)\}$, where m is the biggest element that appears in the views, that is $m = \max_{V \in \boldsymbol{V}} \max_{u \in V} u$.*
3. *For each $i, j \in \{0, \ldots, c-1\}$, for each $V \in \boldsymbol{V}$, for each $u \in V$ such that $u \equiv (j-i)[c]$, there exists $v \in V$ such that:*
   - *there is an edge in H from i to j labeled by $v - u$.*
   - *for each $V' \in \boldsymbol{V}$, for each $u' \in V'$, there exist $v' \in V'$ and an edge in H from i to $(j - v')[c]$ labeled by $(v - u) + (v' - u')$.*

Remark 1. *For a given $\boldsymbol{V}$, the number of nodes and edges of a graph $H \in \mathcal{H}_{\boldsymbol{V}}$ is bounded, thus $\mathcal{H}_{\boldsymbol{V}}$ is finite. Moreover all $H \in \mathcal{H}_{\boldsymbol{V}}$ are complete graphs, because $\boldsymbol{V}$ is C-complete.*

Definition 4 (Weight)**.** *The weight of a path in a graph H is the sum of all labels along edges of the path. A path with no edge is of weight 0.*

Definition 5 (Behavior graph)**.** *Given a C-complete view $\boldsymbol{V}$ such that $C \in \boldsymbol{V}$ with $C = \{c\}$, we define $\mathcal{G}_{\boldsymbol{V}}$ as the set of all directed, edge-labeled graphs G constructed as follows:*

1. *Pick $H \in \mathcal{H}_{\boldsymbol{V}}$, and start with $G = H$. If G only contains cycles of positive weights or cycles of negative weights, skip Step 2 and Step 3. Otherwise, if G contains both cycles of positive and negative weights[1]:*
2. *Pick $i, j \in \{0, \ldots, c-1\}$ such that:*
   - *There exists in G a path from i to j of weight $(i-j)[c]$. Let a be a minimal (in absolute value) weight of a path satisfying this property.*
   - *For all $a' \equiv a[c]$, there exists i', j' such that $(j' - i') \equiv (j - i)[c]$, and there is no edge from i' to j' of label a'.*

   Then, for all i', j' such that $(j' - i') \equiv (j - i)[c]$, add an edge from i' to j' of label a.

[1] Please note that this definition differs slightly from the definition in [4] in that we did not require in [4] that G had both negative and positive cycles in order to proceed to Step 2. This minor difference does not actually change any of the proofs, as Lemma 6 did not make use of this part of the construction, and Lemma 9 already assumes that the graph contains cycles of both signs. It does however make the size analysis at the end of Section 4.1 (which was not featured in [4]) easier.

3. Repeat Step 2 until no more edges can be added.

Remark 2.

- *Step 2 of the construction of $\mathcal{G}_V$ can only be applied a finite number of times for each G, since it can be done at most once for each (i, j). Moreover, there is a finite number of choices at each step. Hence, $\mathcal{G}_V$ is finite.*
- *As soon as there is a path from some i to some j of weight $(i-j)[c]$, then there is a weight $a \equiv (i-j)[c]$ such that all i', j' that are at the same distance as i is from j are linked by an edge of this particular weight, provided that G contains cycles of both positive and negative weights.*

Behavior graphs contain another necessary property of the existence of a path of length n that goes through the x_i's. Remark that a path π of delay $i-j$ going from x_i to x_j is actually of length 0 modulo c. Hence π appears in E as a sequence of C edges. Then the reasoning shown in Figure 5 implies the existence of paths of identical delay from nodes x_{i-u} to nodes x_{j+c-u} for all u.

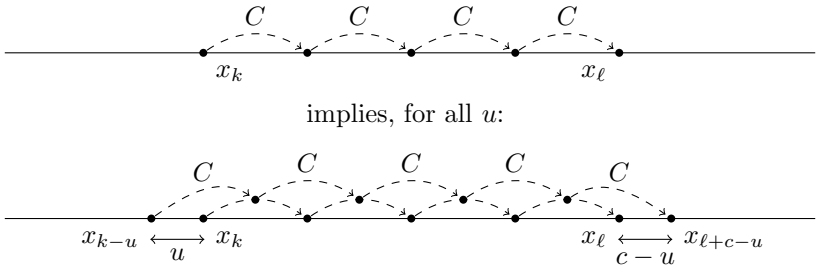

Fig. 5: Illustration of the intuition for the construction of a behavior graph. Assume that the x_i's form a path of length n from x_0 to x_n. If x_k and x_ℓ are connected via a sequence of C's, represented by the dashed edges, then for all $u < c$, there exists some intermediate nodes such that x_{k-u} and $x_{\ell+c-u}$ are connected as shown in the picture.

All the intuitions presented in this section are made precise in Section 4.2 and Section 4.3, when this tool is actually used. Before moving on, we prove two lemmas that give upper bounds on the size of these graphs and the relevant paths they contain, which will be useful for the complexity analysis of the final algorithm.

Lemma 4. *Let G be a weighted graph on nodes $0, \dots, c-1$. Assume that there exists in G a path of weight d going from node i to node j. Then there exists in G a path of weight d going from node i to node j whose length is in $O(c^4 m^4 d + c^6 m^5)$, where m is the weight of greatest absolute value that occurs in G.*

Proof. Let π be a path of weight d going from i to j. By reordering the edges in π, we can assume that π has the following form:

$$
\begin{aligned}
& i \xrightarrow{\rho_0} u_1 \xrightarrow{\theta_{1,1}} u_1 \xrightarrow{\theta_{1,2}} u_1 \xrightarrow{\theta_{1,3}} \cdots \xrightarrow{\theta_{1,k_1}} u_1 \\
& \xrightarrow{\rho_1} u_2 \xrightarrow{\theta_{2,1}} u_2 \xrightarrow{\theta_{2,2}} u_2 \xrightarrow{\theta_{2,3}} \cdots \xrightarrow{\theta_{2,k_2}} u_2 \\
& \xrightarrow{\rho_2} u_3 \xrightarrow{\theta_{3,1}} u_3 \xrightarrow{\theta_{3,2}} u_3 \xrightarrow{\theta_{3,3}} \cdots \xrightarrow{\theta_{3,k_3}} u_3 \\
& \xrightarrow{\rho_3} \cdots \\
& \xrightarrow{\rho_{\ell-1}} u_\ell \xrightarrow{\theta_{\ell,1}} u_\ell \xrightarrow{\theta_{\ell,2}} u_\ell \xrightarrow{\theta_{\ell,3}} \cdots \xrightarrow{\theta_{3,k_\ell}} u_\ell \xrightarrow{\rho_\ell} j
\end{aligned}
$$

In the picture above, i and j are the starting and ending points of the path. The u_s's are pairwise distinct nodes of G, among $\{0, \ldots, c-1\}$. The paths $\theta_{s,k}$ are simple cycles going from u_s to u_s, and the paths ρ_s for $s \in 1, \ldots, \ell - 1$ are simple paths going from u_s to u_{s+1}. Finally ρ_0 and ρ_l are simple paths linking respectively i to u_1 and u_ℓ to j. This shape of path is achieved by selecting (arbitrarily) the position of each line of the picture (at an occurence of a new u_i) and then moving iteratively each simple cycle from u_i to u_i to its correct position in the path, until only simple cycles and paths remain. We also assume that all cycles $\theta_{s,k}$ have non-zero weight, otherwise they can safely be removed without changing the weight of π.

Let $\rho = \rho_0 \cdot \ldots \cdot \rho_\ell$. Since $\ell \leq c$ and each ρ_s is a simple path, then ρ is of length at most $c(c+1)$. Let W be the set of all weights in G, and let $m = \max_{w \in W}(|w|)$. Let w_ρ be the weight of ρ. Then $-mc(c+1) \leq w_\rho \leq mc(c+1)$.

Similarly, let w_θ be the sum of the weights of all $\theta_{s,k}$ cycles. Then $w_\theta + w_\rho = d$. Thus $|w_\theta| \leq |d| + |w_\rho|$. Let Θ be the set of all weights of $\theta_{s,k}$ cycles.

Case 1: Assume that Θ contains only positive or only negative integers. Then, there are at most $|w_\theta|$ occurences of $\theta_{s,k}$ cycles in π. Thus, the total length of π is bounded by $c(c+1)+c|d|+mc^2(c+1)$, which proves the claim.

Case 2: Assume that Θ contains both positive and negative integers. We know that $\gcd(\Theta)$ divides w_θ. Then, by applying Bezout's Identity with positive coefficients, we build a sequence of integers from Θ whose sum is w_θ and whose length is at most $|w_\theta||\Theta|^2 \max(\Theta)^2$, which is bounded by $4(|d| + mc(c+1))(mc)^4$. We deduce from this sequence a new path μ from i to j of weight d by inserting each cycle from Θ used in the sequence at the correct position in ρ. Thus μ is of size at most $4(|d| + mc(c+1))(mc)^4 + c(c+1)$, which ends the proof of the lemma. □

Lemma 5. *Let G be a behavior graph in $\mathcal{G}_{\boldsymbol{V}}$. Then G is of size at most $O(\|\boldsymbol{V}\|^4)$.*

Proof. Let G be a graph in $\mathcal{G}_{\mathbf{V}}$. Then there exists a graph H in $\mathcal{H}_{\mathbf{V}}$ from which G is built, using Definition 5. Now remark that by definition H has c nodes and at most $4mc^2$ edges, where m is the biggest element that appears in any query in $\mathbf{V}$, and with each edge carrying a weight of size at most $4m$. If H only contains positive or negative cycles, then there is nothing else to prove, as G is then of size $O(\|\mathbf{V}\|^4)$.

Otherwise, it remains to consider the number of edges and size of the weights added by Step 2 of Definition 5. First, we notice that Step 2 adds at most c^2 edges, that is, no more than one for each pair of nodes of the graph. Next, we prove that the new weights are bounded in size. Indeed, since there exists at least a cycle of negative weight and a cycle of positive weight, we can deduce that *all* nodes have a cycle of positive weight and a cycle of negative weight, since H is a complete graph. Similarly, we can assume that each node has a positive cycle of weight at most $3u$ and a negative cycle of weight at least $-3u$, where u is the largest absolute value that appears in H. Indeed, as soon as a path's weight becomes larger than u or smaller than $-u$, any edge that returns to its starting point will turn it into a cycle of weight bounded by $-3u$ and $3u$. Finally, these cycles can be used to make sure that no added weight at Step 2 of Definition 5 can be less than $-3cu$ or bigger than $3cu$. All in all, Step 2 of the definition adds at most c^2 which carry weights whose size vary in a range of $6cu = 12cm$. Thus, the added structure is of size $O(\|\mathbf{V}\|^4)$, which ends the proof of the lemma. □

In Section 4.2, we show that, if there exists some behavior graph G such that there is no path of weight 0 from 0 to o in G, then we can build two databases that agree on the view but not on paths of length $n \equiv o[c]$, for all n. In other words, we can build a database whose view satisfies all the necessary conditions for the existence of a path of length n, while still maintaining a non-zero delay between the relevant nodes. On the contrary, in Section 4.3, we show that, if for all behavior graphs G, there is a path of weight 0 from 0 to o in G, then it is enough to satisfy the necessary conditions in order to have a path of length n.

In Section 4.4, we give a precise proof of how our decision algorithm uses the properties of behaviour graphs. The proof can be outlined as follows. For a given C-complete view $\mathbf{V}$ and a given natural number $o \in \{0, \ldots, c-1\}$, we are simply looking for the occurrence of a specific graph $G \in \mathcal{G}_{\mathbf{V}}$, namely one that does not contain a path of weight 0 from node 0 to node o. If we

do find one such G, then, for all $n \equiv o[c]$, $\mathbf{V} \not\twoheadrightarrow \{n\}$. Otherwise, $\mathbf{V} \twoheadrightarrow \{n\}$ for some $n \equiv o[c]$. We do not actually need to compute $\mathcal{G}_{\mathbf{V}}$: we simply guess the appropriate graph G and check that it does contain the critical path. Since G is of size polynomial in $\mathbf{V}$ and the considered path, if it exists, can be assumed to be polynomial in the size of G, our decision algorithm works in PSPACE, more precisely in Π_2^{P}.

4.2 Negative Direction: Building Counter-examples

In this section, we solve the negative case of Proposition 1 by proving the following proposition:

Proposition 2. *Assume that there exists $G \in \mathcal{G}_{\boldsymbol{V}}$ such that there is no path of weight 0 from 0 to o. Then, for all $n \equiv o[c]$, $\boldsymbol{V} \not\twoheadrightarrow \{n\}$.*

The proof of Proposition 2 is split across Lemma 6 and Lemma 9. The canonical counter-examples that we build in order to prove that $\mathbf{V} \not\twoheadrightarrow \{n\}$ depend on whether G only contains cycles of positive or negative weights, or if it actually has both. These two cases are respectively dealt with in Lemma 6 and Lemma 9, and Example 1 and Example 2 give examples of both situations.

Example 1. *Let $\boldsymbol{V} = \{C, V\}$, $C = \{2\}$ and $V = \{1, 2\}$. Figure 6 represents one of the graphs in $\mathcal{G}_{\boldsymbol{V}}$, that additionally satisfies the condition of Lemma 6. Namely, there is no path of weight 0 from 0 to 1, and 0 only has non-negative cycles.*

Example 2. *Let $\boldsymbol{V} = \{C, V\}$, $C = \{3\}$ and $V = \{1, 5\}$. Figure 7 represents one of the graphs in $\mathcal{G}_{\boldsymbol{V}}$, that additionally satisfies the condition of Lemma 9. Namely, there is no path of weight 0 from 0 to 1, and 0 has positive and negative cycles.*

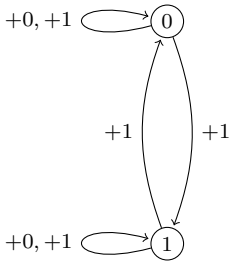

Fig. 6: A behavior graph for the view defined in Example 1.

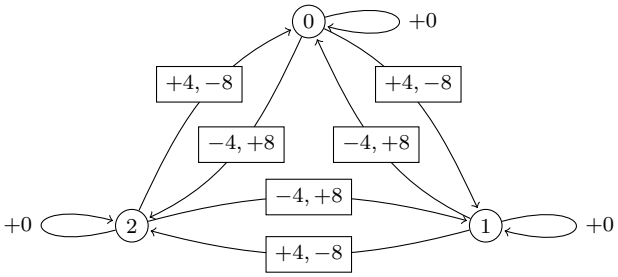

Fig. 7: A behavior graph for the view defined in Example 2.

Lemma 6. *Assume that there exists $G \in \mathcal{G}_{\boldsymbol{V}}$ such that 0 does not have both a cycle of positive weight and a cycle of negative weight and that there is no path of weight 0 from 0 to o. Then, for all $n \equiv o[c]$, $\boldsymbol{V} \not\twoheadrightarrow \{n\}$.*

Proof. Assume that all cycles of 0 in G have positive or zero weight. An example of such a case is given in Example 1.

Let M be the set of all maximal elements of each query in $\mathbf{V}$, that is $M = \{\max(V) \mid V \in \mathbf{V}\}$. Let $d = \gcd(M)$. Remark that d divides c, as c is the maximal (and only) element of C.

Claim 3. *d does not divide o.*

Assume d divides o. Then, Bezout's Identity provides $u_1, \ldots, u_k \in M$ such that $u_1 + \ldots + u_k \equiv o[c]$. Since each u_i belongs to M, then each u_i is the maximal element of some $V_i \in \mathbf{V}$. Then, by construction of G, there exists a path from 0 to o whose weight is of the form $(v_1 - u_1) + (v_2 - u_2) + \ldots + (v_k - u_k)$ where each v_i is an element of V_i. Hence, all terms of the sum are negative or zero. If all are zero, then there is a path of weight zero from 0 to o. Otherwise, there is a path of

negative weight from 0 to o, and by applying the same reasoning from o to 0, we get a cycle of negative weight from 0 to 0. Both these cases are false by assumption, which proves the claim.

For each $V_i \in V$, let u_i be the maximal element of V_i. By construction of d, we know that d divides u_i. Let n be any natural number such that $n \equiv o[c]$. We can now construct a database D as follows:

- D contains two simple paths of length n, whose respective nodes are $x_0, \ldots, x_n$ and $x'_0, \ldots, x'_n$.
- For each $s, t \leq n$ such that $t - s \in V_i$ for some i, and d does not divide $t - s$, we add to D a new path $\pi^i_{s,t}$ of length $u_i - 2$, and we connect x_s and x'_s to its initial node, and we connect its final node to x_t and x'_t.

We then construct D′, a copy of D, in which x_j and x'_j switch roles for each $j \equiv o[d]$. Note that, since d does not divide o, this means that x_0 and x'_0 are not switched, but x_n and x'_n are. See Figure 8 for an example of the construction.

Claim 4. *D and D' agree on $\mathbf{V}$, but each path from x_0 to x_n in D' is strictly longer than n. Hence D and D' disagree on $Q = \{n\}$.*

Let $(x, y) \in V_i(\mathrm{D})$ for some $V_i \in \mathbf{V}$. If either x or y belongs to one of the new paths of the form $\pi^j_{s,t}$, then the symmetry of the construction between the two original simple paths show that $(x, y) \in V_i(\mathrm{D}')$. Otherwise $x = x_s$ or $x = x'_s$ and $y = x_t$ or $y = x'_t$, for some s and t. Then either $t - s \equiv 0[d]$, in which case either both x and y are switched with their copy in D′, or none are. Then, once again the symmetry of the construction concludes that $(x, y) \in V_i(\mathrm{D}')$. Otherwise, d does not divide $t - s$, which implies that (x, y) are linked by $\pi^i_{s,t}$ in V_i, and thus that $(x, y) \in V_i(\mathrm{D}')$. Hence, $\mathbf{V}(\mathrm{D}) = \mathbf{V}(\mathrm{D}')$. Remark now how each path from x_0 to x_n in D′ has to cross one of the $\pi^j_{s,t}$, which is longer than $t - s$. It follows that each path from x_0 to x_n in D′ is longer than n, which proves the claim.

It easily follows from this claim that $\mathbf{V} \not\twoheadrightarrow \{n\}$. The case where 0 only has cycles of negative or zero weight is dealt with in a very similar way, which concludes the proof of the lemma. □

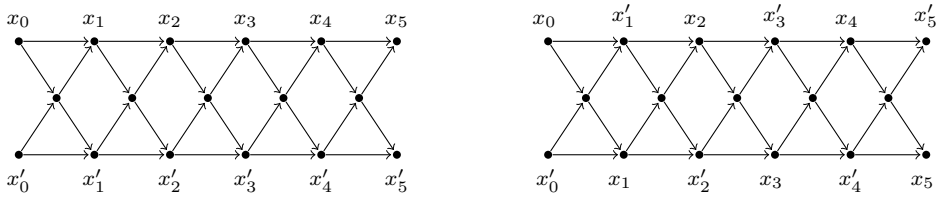

Fig. 8: Example of the construction in Lemma 6 for the view defined in Example 1, that is $\mathbf{V} = \{C, V\}$ with $C = \{2\}$ and $V = \{1, 2\}$. For this example, $d = 2$, $n = 5$ and $o = 1$. Remark how the left and right databases agree on $\mathbf{V}$, while there exists a path of length 5 going from x_0 to x_5 in the left database, but no such path in the right one.

We now move on to the proof of Lemma 9. Note that this proof is a little more involved and mainly relies on the arithmetic properties of behavior graphs, that are given in Lemma 7:

Lemma 7. *Let $G \in \mathcal{G}_{\mathbf{V}}$ such that 0 has both cycles of positive and negative weight. Let W be the set of all weights of cycles of 0 in G, and let $d = \gcd(W)$. Then G has the following properties:*

1. *For all $i, j \in \{0, \ldots, c-1\}$ all paths from i to j have the same weight modulo d. We denote this value by $w(i, j)$. Moreover, $w(i, j)$ is compatible with composition. Namely:*
   - $w(i, i) \equiv 0[d]$
   - *For all k,* $w(i, k) + w(k, j) \equiv w(i, j)[d]$
   - $w(i, j) \equiv -w(j, i)[d]$

2. *For all* $0 \leq i < j < c \wedge d$, $w(i,j) \not\equiv i - j[c \wedge d]$, *where* $c \wedge d$ *denotes* $\gcd(c,d)$.

3. *For all* $i \in \{0, \ldots, c-1\}$, $w(i, i + c \wedge d) \equiv 0[d]$.

Proof.

1. By construction of d, we already know that all cycles of 0 have weight $0[d]$. Let $i, j \in \{0, \ldots, c-1\}$. Let π_1 and π_1' be two paths from i to j of respective weights w_1 and w_1'. Let π_0 be a path from 0 to i of weight w_0 and π_2 be a path from j to 0 of weight w_2. Then both $\pi_0 \cdot \pi_1 \cdot \pi_2$ and $\pi_0 \cdot \pi_1' \cdot \pi_2$ are cycles of 0. Hence, we have $w_0 + w_1 + w_2 \equiv 0[d]$ and $w_0 + w_1' + w_2 \equiv 0[d]$, which implies $w_1 \equiv w_1'[d]$, so that $w(i,j)$ is correctly defined, as in the statement of the lemma.

   The other properties are easy consequences of this fact.

2. To prove this property, we make use of the following claim:

   Claim 5. *For all* $i \in \{0, \ldots, c-1\}$, *for all natural number* k, *there exists* $j \in \{0, \ldots, c-1\}$ *such that* $w(i,j) \equiv -j + k[c \wedge d]$.

   Proof of claim. Let $i \in \{0, \ldots, c-1\}$, let k be any natural number. Since $\mathbf{V}$ is complete, there exists $V, V' \in \mathbf{V}$ such that there exists $u \in V$ and $u' \in V'$ with $u \equiv 1[c]$ and $u' \equiv i - k[c]$. Then, Property 3 of Definition 3 with $i-1$, i, u and u' gives $v \in V$ and $v' \in V'$ such that:

   - $w(i-1, i) \equiv (v - u)[d]$
   - $w(i-1, i-v') \equiv (v-u) + (v' - u')[d]$

   Then it follows that:

$$\begin{aligned}
w(i-1, i-v') &\equiv (v-u) + (v'-u')[c \wedge d] \\
w(i-1, i-v') &\equiv (v-u) + (v'-i+k)[c \wedge d] \\
w(i-1, i) + w(i, i-v') &\equiv (v-u) + (v'-i+k)[c \wedge d] \\
(v-u) + w(i, i-v') &\equiv (v-u) + (v'-i+k)[c \wedge d] \\
w(i, i-v') &\equiv (v'-i) + k[c \wedge d]
\end{aligned}$$

   and we conclude the proof of the claim by renaming $v' - i$ to j. □

   We can now move on to the proof of the property. Assume by contradiction that there exists i, j such that $w(i,j) \equiv i - j[c \wedge d]$. Then $w(i,j) \equiv i - j + kd[c]$ for some k. Since $\gcd(W) = d$ and W contains both positive and negative elements, by using Bezout's Identity with positive coefficients, we show that there exists a cycle of i of weight $-kd$. Hence, there exists a path from i to j of weight $i - j[c]$. Since this path satisfies the requirement in Definition 5, we can show that for all $r \in \{0, \ldots, c-1\}$, $w(r, r + (j-i)) \equiv i - j[c \wedge d]$.

   This is a contradiction with the claim. Indeed, the claim implies that for all k, there exists l such that $w(0,l) \equiv -l + k[c \wedge d]$. Since, k can take $c \wedge d$ different values, but $j - i < c \wedge d$, this means that we can find two values $k \neq k'$ for which the claim produces l and l' that are in the same class modulo $j-i$. More precisely, we have $w(0,l) \equiv -l + k[c \wedge d]$, $w(0,l') \equiv -l' + k'[c \wedge d]$, and $l \equiv l'[j-i]$. Hence, there exists some α such that $l' = l + \alpha(j-i)$. By using what we just proved above α times, we get $w(l,l') \equiv \alpha(i-j)[c \wedge d]$. Hence, $w(l,l') \equiv l - l'[c \wedge d]$. Thus, we have:

$$\begin{aligned}
w(0,l) &\equiv -l + k[c \wedge d] \\
w(0,l) + w(l,l') &\equiv w(l,l') - l + k[c \wedge d] \\
w(0,l') &\equiv l - l' - l + k[c \wedge d] \\
w(0,l') &\equiv -l' + k[c \wedge d]
\end{aligned}$$

   This final equality implies that $k = k'$, which is a contradiction.

3. We first prove that $w(0, 0 + c \wedge d) \equiv 0[c \wedge d]$. This is a purely arithmetic consequence of Property 2, as explained in Lemma 8.

   We rewrite this equality as $w(0, 0 + c \wedge d) \equiv -c \wedge d + kd[c]$. Then, by following the same reasoning as in the proof for Property 2, we prove that there exists a path from 0 to $c \wedge d$ of weight $-c \wedge d[c]$. Since this path satisfies the requirement in Definition 5, then there must exist some weight $w \equiv -c \wedge d[c]$ such that, for all $i \in \{0, \ldots, c-1\}$, there is an edge from i to $i + c \wedge d$ of weight w, which we denote by $\pi_{i,i+c\wedge d}$.

   Let $c' = \frac{c}{c\wedge d}$. Then $\pi_{0,c\wedge d} \cdot \pi_{c\wedge d,2(c\wedge d)} \cdot \ldots \cdot \pi_{(c'-1)(c\wedge d),c'(c\wedge d)}$ is a cycle of 0 of weight $c'w$. Thus, d divides $c'w$. By construction, $c' \wedge d = 1$, hence d divides w, that is, $w \equiv 0[d]$. This implies that for all $i \in \{0, \ldots, c-1\}$, $w(i, i + c \wedge d) \equiv 0[d]$.

□

The proof of Lemma 7 uses the following arithmetical result:

Lemma 8. *Let $d \in \mathbb{N}$, and $a_1, \ldots, a_k \in \{0, \ldots, d-1\}$. Assume that for all i, j, $a_i + a_{i+1} + \ldots + a_j \not\equiv i - j - 1[d]$. Then there are at most $d - k - 1$ possible values for a_{k+1} such that the sequence $a_1, \ldots, a_{k+1}$ also satisfies this property. In particular, if $k = d - 1$, then there are no possible continuations.*

Proof. Assume everything defined as in the statement of the lemma. We define F as:

$$F = \{a \in \{0, \ldots, d-1\} \mid \exists i, a_i + \ldots + a_k + a \equiv i - k - 2[d]\} \cup \{d - 1\}$$

We refer to F as the set of forbidden values for a_{k+1} in the sense that if $a_{k+1} \in F$, then the resulting sequence $a_1, \ldots, a_{k+1}$ does not satisfy the property of the lemma. In the same way, we say that an integer $i \in \{1, \ldots, k\}$ *forbids* a value a when $a_i + \ldots + a_k + a \equiv i - k - 2[d]$. In other words, a is the element of that is obtained for the quantification i in the definition of F. Remark that $d - 1$ belongs to F, as it is always a forbidden value[2] regardless of the remainder of the sequence.

We conclude the proof now by showing that $|F| = k + 1$, which reduces the set of possible values for a_{k+1} to at most $d - k - 1$. This comes from two facts: (1) no $i \in \{1, \ldots, k\}$ forbids $d - 1$ and (2) for all i, j, if $i \neq j$, then i and j forbid two distinct values. We now prove the two facts:

1. Assume that there exists $i \in \{1, \ldots, k\}$ such that i forbids $d - 1$. Then $a_i + \ldots + a_k - 1 \equiv i - k - 2[d]$. Hence, $a_i + \ldots + a_k \equiv i - k - 1[d]$, which is a contradiction.

2. Assume that there exists $j > i$ such that i and j forbid the same value a. Then

   $$a_j + \ldots + a_k + a \equiv j - k - 2[d]$$

   Then

   $$a \equiv j - k - 2 - a_j - \ldots - a_k[d]$$

   But we also have

   $$a_i + \ldots + a_k + a \equiv i - k - 2[d]$$

   Hence

   $$a_i + \ldots + a_{j-1} + j - k - 2 \equiv i - k - 2[d]$$

   Finally

   $$a_i + \ldots + a_{j-1} \equiv i - j[d]$$

   which is a contradiction.

□

We are now ready to give the full proof of Lemma 9:

[2] If $a_i = d - 1$ for some i, then the property of the lemma is falsified by the subsequence a_i obtained with $j = i$.

Lemma 9. *Assume that there exists $G \in \mathcal{G}_{\boldsymbol{V}}$ such that 0 has both cycles of positive and negative weight, and that there is no path of weight 0 from 0 to o. Then, for all $n \equiv o[c]$, $\boldsymbol{V} \not\twoheadrightarrow \{n\}$.*

Proof. Let $G \in \mathcal{G}_{\mathbf{V}}$ be defined as in the statement of the lemma. An example of such a case is given in Example 2. We also define d as in Lemma 7.

Let f be the function defined as follows:

$$\forall i \in \{0, \dots, c \wedge d - 1\},\ f(i) = i + w(0, i)[d]$$

Remark that f is one-to-one. Indeed, assume that i, j are such that $f(i) = f(j)$. Then :

$$\begin{aligned} i + w(0, i) &\equiv j + w(0, j)[d] \\ w(0, i) - w(0, j) &\equiv j - i[d] \end{aligned}$$

And by using Property 1 of Lemma 7 we get:

$$w(j, i) \equiv j - i[d]$$

This final equation implies that $i = j$. Otherwise, this would be a contradiction with Property 2 of Lemma 7. We can then define g as:

$$\forall i \in \{0, \dots, d - 1\},\ g(i) = i + w(0, i)[d]$$

Property 3 of Lemma 7 then gives us:

$$\forall i \in \{0, \dots, d - 1\},\ g(i) = i + w(0, i[c \wedge d])[d]$$

Remark now that g can also be written:

$$\forall i \in \{0, \dots, d - 1\},\ g(i) = f(i[c \wedge d]) + (i - (i[c \wedge d]))[d]$$

from which we deduce that g is one-to-one, because f is. Thus g is a permutation of $\{0, \dots, d - 1\}$.

We can now define two databases D and D′ such that D is a cycle of length d whose nodes are $x_0, \dots, x_{d-1}$, and D′ is a copy of D in which x_i is replaced by $x_{g(i)}$ for all i. This is well defined and also a cycle of length d because g is a permutation. See Figure 9 for an example of this construction.

Claim 6. *D and D' agree on $\boldsymbol{V}$.*

Let x_i and x_j be two nodes of D such that $(x_i, x_j) \in V(\mathrm{D})$ for some $V \in \mathbf{V}$. Then there exists u such that $u \in V$ and $j - i \equiv u[d]$. Let γ be the length of any path from x_i to x_j in D′. Then we have:

$$\begin{aligned} \gamma &\equiv g^{-1}(j) - g^{-1}(i)[d] \\ \gamma &\equiv (j - w(0, j)) - (i - w(0, i))[d] \\ \gamma &\equiv (j - i) - (w(0, j) - w(0, i))[d] \\ \gamma &\equiv u - w(i, j)[d] \end{aligned}$$

By definition of G, there exists $v \in V$ such that $v - u \equiv w(i, j)[d]$. Hence, we have $\gamma \equiv v[d]$. This means that there is a path from x_i to x_j in D′ of length v, and thus $(x_i, x_j) \in V(\mathrm{D}')$. A similar reasoning proves the other direction, and concludes the proof of the claim.

Claim 7. *For all $n \equiv o[c]$, $g(n) \not\equiv n[d]$.*

Assume that $g(n) \equiv n[d]$. Then $w(0, n) \equiv 0[d]$. Hence, Property 3 of Lemma 7 implies that $w(0, o) \equiv 0[d]$. Hence there exists a path in G from 0 to o of weight kd for some k. Since d is the gcd of all cycles of 0, Bezout's Identity implies that there exists a path from 0 to 0 of weight $-kd$. Hence there exists a path from 0 to o of weight 0, which is a contradiction.

It easily follows from the claim that, for all $n \equiv 0[c]$, $\mathbf{V} \not\twoheadrightarrow \{n\}$. Indeed, the only path of length n starting from x_0 ends in x_n in D, whereas it ends in $x_{g(n)}$ in D′. This concludes the proof of the lemma. □

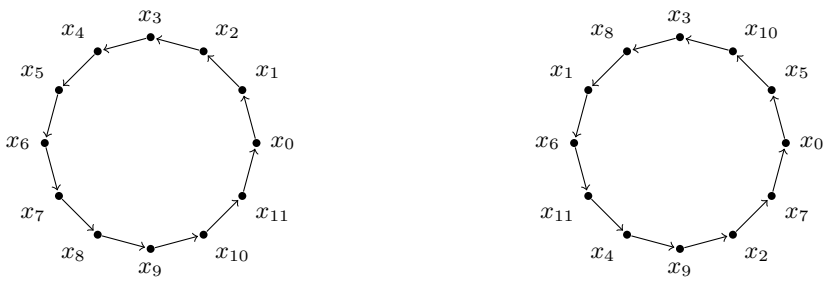

Fig. 9: Example of the construction in Lemma 9 for the view defined in Example 2, that is $\mathbf{V} = \{C, V\}$ with $C = \{3\}$ and $V = \{1, 5\}$. Remark how the left and right databases agree on $\mathbf{V}$, while they disagree on all queries that do not ask for paths of length 0[3].

4.3 Positive Direction: Computing a Rewriting

In this section, we solve the positive case of Proposition 1. We start by giving a simple example that shows some of the features of the rewritings that will be used to prove Proposition 3.

Example 3. *In this example, we work with:*

- $\boldsymbol{V} = \{C, V_1, V_2\}$
- $C = \{2\}$
- $V_1 = \{1, 2\}$
- $V_2 = \{2, 3\}$
- $Q = \{5\}$

We show that $\boldsymbol{V} \twoheadrightarrow Q$. Indeed, the following formula R is a rewriting of Q using $\boldsymbol{V}$.

$$R(x,y) = \exists x_0, \ldots, x_5,\ x_0 = x \wedge x_5 = y \wedge CQ_{\pi_5}(x_0, \ldots, x_5) \wedge \Big(\forall z, V_1(z, x_3) \Rightarrow (C(z, x_3) \vee C(z, x_4))\Big)$$

where π_5 is a simple path whose nodes are $x_0, \ldots, x_5$ and CQ_{π_5} is the conjunctive query that states all the atoms that hold in $\boldsymbol{V}(\pi_5)$, as follows:

$$CQ_{\pi_5}(x_0, \ldots, x_5) = \bigwedge_{0 \leq i \leq 4} V_1(x_i, x_{i+1}) \wedge \bigwedge_{0 \leq i \leq 2} V_2(x_i, x_{i+3}) \\ \bigwedge_{0 \leq i \leq 3} V_1(x_i, x_{i+2}) \wedge V_2(x_i, x_{i+2}) \wedge C(x_i, x_{i+2})$$

First, remark that R only states necessary conditions for the existence of a path of length 5 from x to y, as explained in Section 4.1, hence, for all D, $Q(D) \subseteq R(\boldsymbol{V}(D))$.

Assume now that $(x, y) \in R(\boldsymbol{V}(D))$. Let $x_0, \ldots, x_5$ be a quantification for which $R(x, y)$ is satisfied. We can prove the following:

- *$C(x_0, x_2)$, $C(x_1, x_3)$ and $C(x_2, x_4)$ hold in $\boldsymbol{V}(D)$. Hence, these pairs of nodes are at distance 2 in D.*
- *$V_1(x_4, x_5)$ holds in $\boldsymbol{V}(D)$. Hence, x_4 and x_5 are either at distance 1 or 2. If this distance is 1, then we immediately get a path of length 5 from x_0 to x_5 by using the previous point, as $x_0 \xrightarrow{2} x_2 \xrightarrow{2} x_4 \xrightarrow{1} x_5$.*
- *Similarly, $V_2(x_0, x_3)$ holds in $\boldsymbol{V}(D)$. If the distance from x_0 to x_3 is 3, we immediately get $x_0 \xrightarrow{3} x_3 \xrightarrow{2} x_5$. Otherwise, there exists z such that $x_0 \to z \to x_3$. This implies $V_1(z, x_3)$.*
- *The remaining case is represented in Figure 10, with the two possible implications of $V_1(z, x_3)$ given by R. Both possibilities also imply a path of length 5 from x_0 to x_5.*

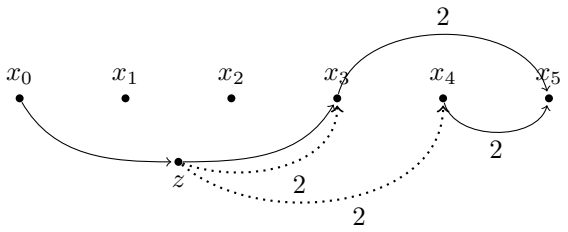

Fig. 10: Illustration for the last case of Example 3. The full edges represent paths in the database, along with their length when it is more than 1. The dotted edges represent the two possible implications of $V_1(z, x_3)$ given by R.

Proposition 3. *Assume that for all $G \in \mathcal{G}_{\boldsymbol{V}}$ there is a path of weight 0 from 0 to o. Then there exists a computable $n \equiv o[c]$ of size $2^{O(\|V\|^4)}$ such that $\boldsymbol{V} \twoheadrightarrow \{n\}$ and we can effectively compute a first-order rewriting that witnesses it.*

Proof. For each $G \in \mathcal{G}_{\mathbf{V}}$, let π_G be the shortest path in G of weight 0 from 0 to o. Let ρ_G be the longest path that is used to build G from some $H \in \mathcal{H}_{\mathbf{V}}$, in Step 2 of Definition 5 and k_G be the number of iterations of Step 2 used to build G from H. Let $k = \max_{G \in \mathcal{G}_{\mathbf{V}}} k_G$. Let $\rho = \max_{G \in \mathcal{G}_{\mathbf{V}}} |\rho_G|$. Let $K = \rho^k$. Let $L = \max_{G \in \mathcal{G}_{\mathbf{V}}} |\pi_G|$. Let $M = 2ck + 3cm$ where m is the biggest number that occurs in one of the views. Let $N = |\mathcal{H}_{\mathbf{V}}|$. Let $n' = K \cdot L \cdot M \cdot N$. Let n be the smallest number such that $n \equiv o[c]$ and $n \geq n'$.

Using Definition 3, Definition 5, Lemma 4 and Lemma 5, we can check that:

- K is in $O(\|\mathbf{V}\|^{18\|\mathbf{V}\|})$;
- L and M are polynomial in $\|\mathbf{V}\|$, being respectively $O(\|\mathbf{V}\|^{18})$ and $O(\|\mathbf{V}\|^2)$;
- N is of size $2^{O(\|\mathbf{V}\|^4)}$, as it is the set of all behavior graphs for $\mathbf{V}$, and each is of size $O(\|\mathbf{V}\|^4)$.

Thus n is in $2^{O(\|\mathbf{V}\|^4)}$.

Claim 8. *$\boldsymbol{V}$ determines $Q = \{n\}$.*

Let R_1 be the $(n+1)$-ary conjunctive query deduced from $\mathbf{V}(x_0 \dots x_n)$, with $x_0, \dots, x_n$ as free variables. It simply states all the atoms that hold in the view of the simple path $x_0 \dots x_n$, as follows:

$$R_1(x_0, \dots, x_n) = \bigwedge_{V \in \mathbf{V}} \bigwedge_{\substack{i,j \\ (x_i, x_j) \in V(x_0 \dots x_n)}} V(x_i, x_j)$$

We also define:

$$R_2(x_0, \dots, x_n) = \forall z, \bigwedge_{i=0}^{n} \bigwedge_{V \in \mathbf{V}} V(z, x_i) \Rightarrow \bigvee_{j=0}^{c} \bigvee_{\substack{u \in V \\ u \equiv j[c]}} V_c^{\frac{u-j}{c}+1}(z, x_{i+c-j})$$

and:

$$R_3(x_0, \dots, x_n) = \bigwedge_{i,j=0}^{n} \bigwedge_{k=1}^{\frac{n+A}{c}} V_c^k(x_i, x_j) \Rightarrow \bigwedge_{l=0}^{c-1} V_c^{k+1}(x_{i-l}, x_{j+c-l})$$

where A is the biggest weight that occurs in a graph in $\mathcal{G}_{\mathbf{V}}$.

Finally, we define R as:

$$\mathrm{R}(x, y) = \exists x_0, \dots, x_n,\ x_0 = x \wedge x_n = y \wedge \bigwedge_{i=1}^{3} R_i(x_0, \dots, x_n)$$

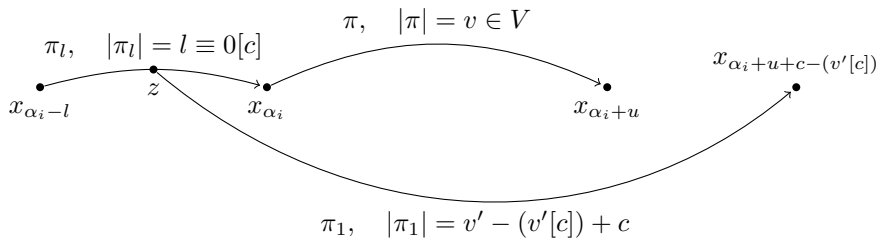

Fig. 11: Illustration for the various notations in the definition of H_r. An additional, undrawn, information is that the path from z to x_{α_i+u} along the drawn edges is of length u'.

Intuitively, each R_i enforces a necessary condition for the existence of a path of length n going from $x = x_0$ to $y = x_n$, as explained in Section 4.1. R_1 and R_2 test conditions $(C2)$ and $(C3)$ respectively, while R_3 enforces the additional constraint of behavior graphs in Definition 5, as illustrated on Figure 5.

It now remains to show that these necessary conditions actually become sufficient under the hypotheses of the proposition. Thus, we can rephrase Claim 8 as:

Claim 9. *R is a rewriting of $Q = \{n\}$ with regards to $\mathbf{V}$.*

Let D be a database, and x and y be two distinguished nodes of D. Assume that $(x, y) \in \mathrm{Q}(\mathrm{D})$. Then there exists a path of length n from x to y. Let $x_0, \ldots, x_n$ be the $n+1$, possibly repeating, nodes of this path. Then it is easy to check that $x_0, \ldots, x_n$ satisfy R_1, R_2 and R_3 in $\mathbf{V}(\mathrm{D})$. Hence, $R(x, y)$ holds in $\mathbf{V}(\mathrm{D})$.

Conversely, assume that $R(x, y)$ holds in $\mathbf{V}(\mathrm{D})$. There exists $x_0, \ldots, x_n$ such that $x = x_0$, $y = x_n$, and $R_1(x_0, \ldots, x_n)$, $R_2(x_0, \ldots, x_n)$ and $R_3(x_0, \ldots, x_n)$ all hold in $\mathbf{V}(\mathrm{D})$. We define, for all $r < K \cdot L \cdot N$:

$$p_r = \{x_{r \cdot M + ck}, x_{r \cdot M + ck + 1}, \ldots, x_{r \cdot M + ck + 3cm} = x_{(r+1) \cdot M - ck}\}$$

Each p_r is a set of $3cm + 1$ consecutive nodes among the x_i's, and all p_r's are disjoint. Additionally, for all paths π in D from some x_i to some x_j, we define $\delta(\pi) = |\pi| - (j - i)$. To each p_r we associate[3] a directed edge-labeled graph H_r defined as follows:

- H_r has c nodes which we will simply note $0, 1, \ldots, c-1$.
- For all $i \in \{0, \ldots, c-1\}$, let $\alpha_i = r \cdot M + ck + i + 2cm$. Remark that $\alpha_i \equiv i[c]$. Then, for all $V \in \mathbf{V}$, for all $u \in V$:
  - We pick a path π in D from x_{α_i} to x_{α_i+u} that satisfies V. There exists one, because $R_1(x_0, \ldots, x_n)$ holds. Let $v = |\pi|$. We add to H_r an edge from i to $i + u[c]$ labeled $\delta(\pi) = v - u$.
  - For all $V' \in \mathbf{V}$, for all $u' \in V'$, let l be the smallest number such that $l + v \geq u'$ and $l \equiv 0[c]$. Let π_l be a path of D of length l from x_{α_i-l} to x_{α_i}. There exists one because $R_1(x_0, \ldots, x_n)$ holds. Let $\pi_0 = \pi_l \cdot \pi$. Let z be the u'th predecessor of x_{α_i+u} along this path. Then $V'(z, x_{\alpha_i+u})$ holds. Then R_2 implies that there exists $v' \in V'$ and a path π_1 in D from z to $x_{\alpha_i+u+c-(v'[c])}$ of length $v' - (v'[c]) + c$. Hence, there is a path π' from x_{α_i-l} to $x_{\alpha_i+u+c-(v'[c])}$ of length $((l + v) - u' + (v' - (v'[c]) + c))$. We then add to H_r an edge from i to $i + u - v'[c]$ labeled $\delta(\pi') = (v - u) + (v' - u')$.

See Figure 11 for a visual representation of the various notations.

[3] arbitrarily, when there are more than one possibility.

Claim 10.

- *For all r, $H_r \in \mathcal{H}_{\boldsymbol{V}}$.*
- *For all r, for all edge of label a from i to j in H_r, there exists $x_{i'}$ and $x_{j'}$ in p_r such that there exists a path π in D from $x_{i'}$ and $x_{j'}$ with $\delta(\pi) = a$, $i' \equiv i[c]$ and $j' \equiv j[c]$.*

Since there are $K \cdot L \cdot N$ different p_r's, then there are at least $K \cdot L$ of them that are attributed the same graph $H \in \mathcal{H}_{\mathbf{V}}$. This means that there exists an increasing function f such that, for all $r < K \cdot L$, $p_{f(r)}$ is attributed H. For ease of notations, we rename $p_{f(r)}$ to q_r^0 and H to G_0. Let $G_0, \dots, G_k$ be k successive iterations as described in Definition 5 on G_0. Then $G_k \in \mathcal{G}_{\mathbf{V}}$ and for each s, G_{s+1} is deduced from G_s by doing one iteration of Step 2 in Definition 5.

Claim 11. *For all $s \in \{0, \dots, k\}$, there exist $\frac{K \cdot L}{\rho^s}$ disjoint sets q_r^s that consist of consecutive nodes among the x_i's such that:*

1. *The distance between q_r^s's last index and q_{r+1}^s's first index is at least $2c(k-s)$. Additionally, the first index of q_0^s is at least $c(k-s)$ and the last index of $q_{\frac{K \cdot L}{\rho^s}}^s$ is at most $n' - c(k-s)$.*
2. *For all r, for all edge of label a from i to j in G_s, there exists $x_{i'}$ and $x_{j'}$ in q_r^s such that there exists a path π in D from $x_{i'}$ and $x_{j'}$ with $\delta(\pi) = a$, $i' \equiv i[c]$ and $j' \equiv j[c]$.*

We prove this claim by induction on s. For $s = 0$, the correctly named q_r^0's already satisfy the required properties.

Assume that, at Step s, the properties are true for some q_r^s's. For each $l < \frac{K \cdot L}{\rho^{s+1}}$, let α_l be the first index of $q_{l\rho}^s$ and β_l be the last index of $q_{(l+1)\rho-1}^s$. Then we define q_l^{s+1} as $q_l^{s+1} = \{x_{\alpha_l - c}, \dots, x_{\beta_l + c}\}$.

It is easy to see that the q_l^{s+1}'s defined as such satisfy Property 1 and also Property 2 for the edges of G_{s+1} that are already in G_s. Let μ be the path in G_s that is used to build G_{s+1} by applying Step 2. Then μ is a path of label a and of length $\eta \leq \rho$ and we have $\mu = i_0 a_1 i_1 \dots i_{\eta-1} a_\eta i_\eta$, where, for all t, there is an edge of label a_t from i_t to i_{t+1} in G_s. Then Property 2 applied at Step s implies that for all t there exists a two nodes $x_{i'_t}$ and $x_{i''_{t+1}}$ in $q_{l\rho+t}^s$ such that there exists a path π_t from $x_{i'_t}$ to $x_{i''_{t+1}}$ with $\delta(\pi_t) = a_t$, $i'_t \equiv i_t[c]$ and $i''_{t+1} \equiv i_{t+1}[c]$.

Since $R_1(x_0, \dots, x_n)$ holds, and for all t $i''_t \equiv i'_t[c]$, then there exists a path π'_t from i''_t to i'_t with $\delta(\pi'_t) = 0$. Hence, we can define π as $\pi = \pi_0 \cdot \pi'_0 \cdot \pi_1 \dots \pi_{\eta-2} \cdot \pi'_{\eta-2} \pi_{\eta-1}$. π is a path from $x_{i'_0}$ to $x_{i''_\eta}$ with $\delta(\pi) = \sum a_t = a$, $i'_0 \equiv i_0[c]$ and $i''_\eta \equiv i_\eta[c]$. Then Property 2 is true for all edges of G_{s+1} from i_0 to i_η.

Since $\delta(\pi) = a$, then $\delta(\pi) \leq A$. Hence $|\pi| \leq n + A$. Hence, we can apply $R_3(x_0, \dots, x_n)$ and get the other required paths. This ends the proof of the claim.

Finally, the claim applied for $s = k$ proves that there exists L sets q_r^k of consecutive x_i's that satisfy property 1 and 2 for some $G_k \in \mathcal{G}_{\mathbf{V}}$. By hypothesis, there exists a path in G_k from 0 to o of length at most L and of weight 0. We conclude by applying once more the reasoning in the proof of the claim. We deduce that there exists two nodes x_i and x_j such that there exists a path π in D with $\delta(\pi) = 0$, $i \equiv 0[c]$ and $j \equiv o[c]$. We complete π with a path π_1 from x_0 to x_i and a path π_2 from x_j to x_n with $\delta(\pi_1) = \delta(\pi_2) = 0$ that are provided by $R_1(x_0, \dots, x_n)$. Hence $\pi_1 \cdot \pi \cdot \pi_2$ is a path from x_0 to x_n with $\delta(\pi_1 \cdot \pi \cdot \pi_2) = 0$. Thus, the length of $\pi_1 \cdot \pi \cdot \pi_2$ is n, and $(x, y) \in \mathrm{Q(D)}$. This ends the proof of the proposition. □

4.4 Decision Algorithm

In this section, we finally prove Proposition 1 by piecing together the tools from Section 4.1 and the results from Section 4.2 and Section 4.3. Recall the statement of Proposition 1:

Proposition 1. *Given a C-complete view $\boldsymbol{V}$ defined by unions of path queries, such that $C \in \boldsymbol{V}$ with $C = \{c\}$ for some $c \in \mathbb{N}$ and a natural number $o \in \{0, \dots, c-1\}$, it is decidable in Π_2^P whether there exists a query $Q = \{n\}$ such that $n \equiv o[c]$ and $\boldsymbol{V} \twoheadrightarrow Q$. If this is the case, such a query Q of size $2^{O(\|\boldsymbol{V}\|^4)}$ and a first-order rewriting of Q with regards to $\boldsymbol{V}$ can be computed effectively.*

Proof. Let $\mathbf{V}$ be a C-complete view defined by unions of path queries, such that $C \in \mathbf{V}$ with $C = \{c\}$ for some $c \in \mathbb{N}$. Let $o \in \{0, \dots, c-1\}$. Assume that there exists a behavior graph $G \in \mathcal{G}_{\mathbf{V}}$ such that G contains no path of weight 0 going from its 0 node to its o node. Then Proposition 2 proves that there exists no query $\mathrm{Q} = \{n\}$ with $n \equiv o[c]$ such that $\mathbf{V} \twoheadrightarrow \mathrm{Q}$. Conversely, assume that all $G \in \mathcal{G}_{\mathbf{V}}$ do contain a path of weight 0 going from their 0 nodes to their o nodes. Then Proposition 3 effectively computes a query $\mathrm{Q} = \{n\}$ with $n \equiv o[c]$ such that $\mathbf{V} \twoheadrightarrow \mathrm{Q}$ and a first-order rewriting of Q using $\mathbf{V}$. Thus, Proposition 1 reduces to the following claim:

Claim 12. *It is decidable in Π_2^P whether all behavior graphs $G \in \mathcal{G}_{\boldsymbol{V}}$ contain a path of weight 0 going from their 0 nodes to their o nodes.*

The statement of the claim naively translates to a Π_2^{P} algorithm, whose universal part considers all graphs G of the correct form and size and whose existential part tests either that $G \notin \mathcal{G}_{\mathbf{V}}$ or that G contains a path of weight 0 from node 0 to node o. To ensure that it is correct, it remains to check that (1) all graphs in $\mathcal{G}_{\mathbf{V}}$ are of polynomial size in the size of the queries in $\mathbf{V}$, (2) it can be checked in NP whether a graph does not belong to $\mathcal{G}_{\mathbf{V}}$ and (3) it can be checked in NP whether a graph contains a path of weight 0 from node 0 to node o.

We will actually substitute (2) with a weaker property (2'), that only tests whether a graph of the correct form and size does not satisfy the second bullet of Remark 2, instead of the whole construction of Definition 5. Indeed, since the proof of Proposition 2 does not actually rely on the exact construction of $\mathcal{G}_{\mathbf{V}}$ but only on the property stated in Remark 2, it is actually equivalent to prove the claim for any extension of choice graphs that contains $\mathcal{G}_{\mathbf{V}}$ and satisfies Remark 2.

Property (1) is a direct application of Lemma 5. It turns out that (2') and (3) are actually consequences of Lemma 4:

(2') Let G be a weighted graph of polynomial size in the size of the queries in $\mathbf{V}$. Checking that G is an extension of a choice graph can be done in polynomial time by simply going through all edges of G and ensuring that all conditions of Definition 3 are satisfied. Then, checking that G does not satisfy Remark 2 is done by non deterministically guessing the existence of cycles of positive and negative weights (of polynomial size, as explained in the proof of Lemma 4), a pair of nodes i, j that are not linked by a path of weight $(i-j)[c]$ and then guessing another pair of nodes i', j' such that $i' - j' \equiv i - j[c]$ and a path of weight $(i-j)[c]$ going from i' to j'. This final path can correctly be assumed to be of polynomial size if it exists, thanks to Lemma 4.

(3) This is an immediate consequence of Lemma 4. Given a weighted graph G, we non deterministically guess a path of weight 0 going from node 0 to node o. This path can be assumed to be of polynomial size in the size of G if it exists, thanks to Lemma 4. This last item concludes the proof of the proposition.

□

5 Extensions

In this section, we discuss several possible extensions of the work presented here. More precisely, Section 5.1 shows the issue that remains to be solved in order to go from asymptotic determinacy to general determinacy, while Section 5.2 considers extensions to stronger view and query languages.

5.1 The Case of Small Queries

This section is devoted to producing the full determinacy picture for the view below. As the asymptotic determinacy picture of this view is already known thanks to Theorem 1, this should highlight what remains to be done in order to decide general determinacy. In all this section, we consider the following view:

- $\mathbf{V} = \{C, V_1, V_2\}$
- $V_1 = \{1, 2\}$
- $C = \{2\}$
- $V_2 = \{2, 5\}$

Claim 13. *For all even n, $\boldsymbol{V} \twoheadrightarrow Q = \{n\}$. This easily comes from $C = \{2\}$.*

By applying Theorem 1 we can show that there exists some odd n such that $\mathbf{V} \twoheadrightarrow \mathrm{Q} = \{n\}$, hence $\mathbf{V}$ also determines all bigger queries. In order to get the full picture, we need to find the smallest odd n that is determined by $\mathbf{V}$. Our work so far actually gives us:

Claim 14. *For all odd $n \leq 7$, $\boldsymbol{V} \not\twoheadrightarrow Q = \{n\}$.*

To prove this claim, we use a technique that is very similar to Lemma 6. More precisely, the two databases in Figure 12 agree on $\mathbf{V}$, but disagree on all $\mathrm{Q} = \{n\}$ when n is odd and not greater than 7.

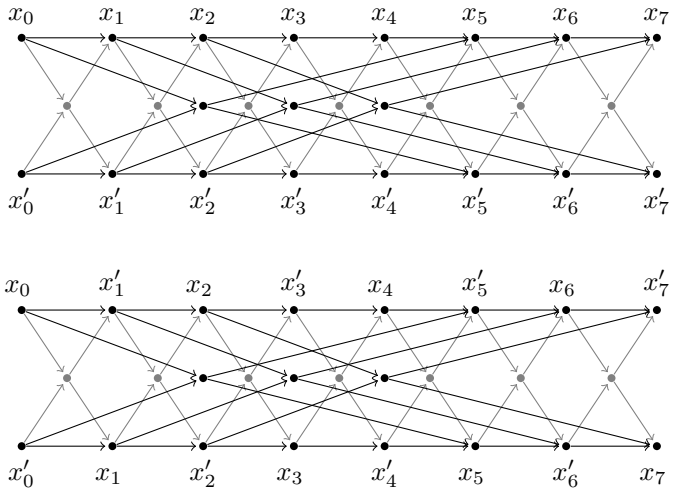

Fig. 12: The two databases above are a proof that $\mathbf{V} \not\twoheadrightarrow \mathrm{Q} = \{n\}$ for any odd n that is not greater than 7. Indeed, we can check that both databases agree on $\mathbf{V}$. However, there is no path of length 1 (respectively 3, 5 and 7) from x_0 to x_1 (respectively x_3, x_5 and x_7) in the bottom database.

Note that this technique does not work for n greater than 7. Indeed, in the case shown above, any path that goes from x_0 to x_7 in the bottom database has to cross from the top section to the bottom section. By doing so, it suffers a delay of either $+1$ or -3 compared to the expected value. It works here because 7 is "too small" and does not provide enough space to catch-up on this delay. Assume now that $n = 9$, then a delay of -3 can be mitigated by following a $+1$ path three times, and thus does not provide a counter-example.

Claim 15. *For all $n \geq 11$, $\boldsymbol{V} \twoheadrightarrow Q = \{n\}$.*

We show this by arguing that $\mathbf{V} \twoheadrightarrow \mathrm{Q} = \{11\}$. This is done by actually proving that the canonical rewriting R given in Section 4.3 works in this case. Although the proof given in Section 4.3 does not apply (because 11 is not "big enough" for all the combinatorial arguments to go through), a careful enumeration of all the possibilities for a database satisfying R actually shows that $\mathrm{R}(x, y)$ implies a path of length 11 from x to y, as was done in Example 3.

It is then straightforward to prove that $\mathbf{V}$ determines every odd query bigger than 11. Let $n = 11 + 2k$ be such a query. Then a rewriting for n is simply $\mathrm{R}_{11} \cdot C^k$, as in Lemma 3. As we already know that $\mathbf{V}$ determines every even query, this end the proof of the claim.

The case of $n = 9$. There remains only a single unsolved case, which is $n = 9$. This qualifies as a "small query" for the view $\mathbf{V}$: a query for which we are unable to either build a generic counter-example, as in Section 4.2, or provide a generic rewriting, as in Section 4.3. We actually proved that $\mathbf{V} \not\twoheadrightarrow \mathrm{Q} = \{9\}$. However, the smallest counter-example that we know is a pair of databases of 154 nodes each, that were built by hand through a very tedious trial and error process and checked by a computer program. At this time, we are unfortunately unable to provide any technique to generate such a counter-example for other views and queries. We conjecture that the combinatorial complexity of these "small queries" might be much higher than what we have dealt with so far.

5.2 Language Extensions

In this section, we show how to extend parts of our work to stronger query languages. We do so by extending the view and query languages in two directions: first, we allow the unions in the view to be arbitrary, that is, we allow the sets associated with the queries in the view to be infinite. Second, we allow the database schema to contain more than one symbol and adapt the query and view languages accordingly. To this end, a natural extension is to consider queries defined by chain queries and views defined by unions of chains queries. In [2], chain queries are defined as conjunctive queries whose underlying graph is a directed path. Here, we provide an equivalent definition that more closely resembles the way we defined path queries.

A *chain query* Q is defined by a single word $w \in \sigma^*$. On a given database D, Q returns all pairs of nodes (x, y) in D such that there exists a path in D of the form $xa_0x_1a_1 \dots x_{m-1}a_{m-1}y$, with $a_0 \dots a_{m-1} = w$. The word $a_0 \dots a_{m-1}$ is called the *label* of π and is denoted $\lambda(\pi)$. Thus, $\mathrm{Q(D)} = \{(x, y) \in \mathrm{D} \mid \exists \pi, x \xrightarrow{\pi} y \text{ and } \lambda(\pi) = w\}$. As with path queries, we will simply note $\mathrm{Q} = \{w\}$.

Similarly, an *arbitrary union of chain queries* Q is defined by a (possibly infinite) set of words $\{w_1, \dots, w_n, \dots\}$ of σ^*. It returns all pairs of nodes connected by a path whose label falls into this set, that is, $\mathrm{Q(D)} = \{(x, y) \in \mathrm{D} \mid \exists \pi, x \xrightarrow{\pi} y \text{ and } \lambda(\pi) \in \mathrm{Q}\}$. As with unions of path queries, we can remark that unions of chain queries generalize chain queries, in the sense that a chain query is simply a union of chain queries whose associated set is of size 1.

Remark that arbitrary unions of chain queries correspond to "any language" path queries, in that they contain all query languages that select pairs of nodes based on the label of a path linking them, such as regular path queries, context-free path queries, and so on.

Remark also that we do not have any theoretical requirement on the way in which the infinite sets associated with these queries should be represented. Indeed, we will shortly see that when an arbitrary union-of-paths view determines a path query, then its finite component already determines the same query. However, for the following construction to be effective, we do require:

- the ability to decide, given a query, whether its associated set is infinite.
- the ability to effectively list all the elements in the associated set, when it is finite.

The following lemma gives a formal statement to the intuition that infinite unions cannot be used to determine a path query.

Lemma 10. *Let Q be a chain query and $\boldsymbol{V}$ be a view defined by arbitrary unions of chain queries. Let $\boldsymbol{V} = \boldsymbol{V}_f \uplus \boldsymbol{V}_\infty$, such that $\boldsymbol{V}_f$ only contains queries defined by finite sets, and $\boldsymbol{V}_\infty$ only contains queries defined by infinite sets. Then $\boldsymbol{V} \twoheadrightarrow Q$ if and only if $\boldsymbol{V}_f \twoheadrightarrow Q$.*

Proof. It is easy to see that if $\mathbf{V}_f \twoheadrightarrow \mathrm{Q}$, then $\mathbf{V} \twoheadrightarrow \mathrm{Q}$. Conversely, assume that $\mathbf{V}_f$ does not determine Q. Then there exists two databases D_1 and D_2 such that D_1 and D_2 agree on $\mathbf{V}_f$ but not on Q. Let k be the length of the longest word that appears in $\mathrm{Q} \cup \mathbf{V}_f$. We transform D_1 into a new database D'_1 as follows:

- We add to D'_1 $k + 1$ new nodes $x_0, \dots, x_k$, as well as the following edges:
  - For all i and for all $a \in \sigma$, $a(x_i, x_{i+1})$ holds in D'_1.

  - For all $a \in \sigma$, $a(x_0, x_0)$ and $a(x_k, x_k)$ hold in D'_1.

- For each original node x of D_1 and all $a \in \sigma$, we add $a(x, x_0)$ and $a(x_k, x)$ to D'_1.

We then apply the same steps to D_2 and get a new database D'_2. This construction has no effect on Q or $\mathbf{V}_f$ for the original nodes of D_1 and D_2. However, for each $(x, y) \in \mathrm{D}_1$ (respectively D_2) and for each $V \in \mathbf{V}_\infty$, $V(x, y)$ holds in $\mathbf{V}_\infty(\mathrm{D}'_1)$ (respectively $\mathbf{V}_\infty(\mathrm{D}'_2)$). Thus, we can check that D'_1 and D'_2 agree on $\mathbf{V}$ but not on Q. Hence $\mathbf{V} \not\twoheadrightarrow \mathrm{Q}$, which concludes the proof. □

In view of Lemma 10, we can restrict our attention to views defined by finite unions of chain queries. We now show how to translate the necessary conditions from Section 3.1 for a view $\mathbf{V}$ defined by finite unions of chain queries to determine a chain query Q. We start by giving an analogue to Lemma 1, by showing that if $\mathbf{V} \twoheadrightarrow \mathrm{Q}$, then $\mathbf{V}$ contains a chain query. We can say even more: each edge in the graph representation of Q must actually belong to a path that satisfies some chain query in $\mathbf{V}$. While this was the case in the previous setting, it was also trivial then.

Lemma 11. *Let $Q = \{w\}$ be a chain query, and $\boldsymbol{V}$ be a view defined by unions of chain queries. Let D be the database consisting of a simple path $\pi = x_0 a_0 x_1 \dots x_{n-1} a_{n-1} x_n$ such that $\lambda(\pi) = w$. Assume that $\boldsymbol{V} \twoheadrightarrow Q$. Then for each $0 \leq i < n$, there exists $j \leq i \leq k$ such that $C = \{a_j \cdot \ldots \cdot a_k\} \in \boldsymbol{V}$.*

Proof. This proof is an extension of the proof of Lemma 1. Assume by contradiction that there exists $0 \leq i < n$ such that the edge $x_i \xrightarrow{a_i} x_{i+1}$ of D is not used to satisfy any chain query of $\mathbf{V}$. In other words, there is no chain query $C \in \mathbf{V}$ such that $(x_j, x_k) \in C(\mathrm{D})$ for any $j \leq i$ and $k > i$.

We now build a database D_1 as follows:

- D_1 contains the simple path π.
- For each $j \leq i$ and $k > i$ such that there exists $V \in \mathbf{V}$ with $a_j \dots a_k \in V$, we add to D_1 a simple path $\pi_{j,k,V}$ from x_j to x_k such that $\lambda(\pi_{j,k,V}) \in V - \{a_j \dots a_k\}$. Such a path exists because V is not a chain query, by our hypothesis.

We then build a database D_2 that is a copy of D_1 except that $a_i(x_i, x_{i+1})$ does not hold in D_2. Then, it remains to check that $\mathbf{V}(\mathrm{D}_1) = \mathbf{V}(\mathrm{D}_2)$ as was the case in the proof of Lemma 1, and that $\mathrm{Q}(\mathrm{D}_1) \neq \mathrm{Q}(\mathrm{D}_2)$. Thus $\mathbf{V} \not\twoheadrightarrow \mathrm{Q}$, which concludes the proof. □

This lemma greatly restricts the form of chain queries Q that can possibly be determined by a given view $\mathbf{V}$. Indeed, a chain query $\mathrm{Q} = \{w\}$ can only be determined by a view $\mathbf{V}$ if w consists of (possibly overlapping) words taken from the chain queries in $\mathbf{V}$ and stitched together. Remark that what makes the problem non-trivial is precisely this overlapping, in the same way that a view $\mathbf{V}$ defined by unions of path queries can determine a path query Q that is not a multiple of the necessary path query in $\mathbf{V}$.

Next, we remark that Lemma 3 immediately applies in this setting. Indeed, if $\mathbf{V} \twoheadrightarrow \mathrm{Q}$, then $\mathbf{V}$ necessarily contains a chain query C. Then, we can deduce that $\mathbf{V} \twoheadrightarrow \mathrm{Q} \cdot C^k$, for any k.

We still however have one major challenge to overcome in order to adapt the remainder of our proof to this extended setting comes from the fact that, when the alphabet is reduced to a single symbol, the resulting free monoid is commutative. This implicit argument is crucial both for the alternate definition of asymptotic determinacy of Section 3.2 and for the arithmetic proofs of Section 4. It is likely that the case of chain queries behaves differently and requires new techniques and ideas. We leave this interesting question for future work.

6 Conclusions

We have shown that, given a view $\mathbf{V}$ defined by unions of path queries, we can decide determinacy of almost all path queries Q. Although the smallest query that we can handle is of exponential size in the size of $\mathbf{V}$, our decision procedure still works with Π_2^{P} complexity. Moreover, for the queries

that are big enough to be handled by our algorithm, we also provide a first-order rewriting when they are determined, and a canonical counter-example otherwise.

A natural continuation of this work would be to try and solve the determinacy problem even for small queries. Another possible continuation stems from the following remark: on all examples where $\mathbf{V}$ determines Q that we are aware of, it also turns out that our rewriting is actually correct, even when the query is too small to be handled by our technique. Perhaps it so happens that this rewriting is always correct, as soon as we assume that $\mathbf{V}$ determines Q. Failing that, it might still be the case that a first-order rewriting can always be found.

Acknowledgements I gratefully thank Luc Segoufin and Cristina Sirangelo for carefully proof-reading this paper and providing many invaluable comments and advices. I also thank the anonymous reviewers for their painstaking and helpful reviews.

References

[1] Serge Abiteboul and Oliver M. Duschka. Complexity of answering queries using materialized views. In *ACM Symp. on Principles of Database Systems (PODS)*, pages 254–263, 1998.

[2] Foto N. Afrati. Determinacy and query rewriting for conjunctive queries and views. *Theoretical Computer Science*, 412(11):1005–1021, 2011.

[3] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi. Lossless regular views. In *ACM Symp. on Principles of Database Systems (PODS)*, pages 247–258. ACM, 2002.

[4] Nadime Francis. Asymptotic determinacy of path queries using union-of-paths views. In *18th International Conference on Database Theory (ICDT 2015)*, 2015.

[5] Nadime Francis, Luc Segoufin, and Cristina Sirangelo. Datalog rewritings of regular path queries using views. *Logical Methods in Computer Science*, 11(4), 2015.

[6] Tomasz Gogacz and Jerzy Marcinkowski. The hunt for a red spider: Conjunctive query determinacy is undecidable. In *Proceedings of the 2015 30th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 281–292. IEEE Computer Society, 2015.

[7] Tomasz Gogacz and Jerzy Marcinkowski. Red spider meets a rainworm: Conjunctive query finite determinacy is undecidable. *arXiv preprint arXiv:1512.01681, to appear in PODS 2016*, 2015.

[8] Alon Y. Levy, Alberto O. Mendelzon, Yehoshua Sagiv, and Divesh Srivastava. Answering queries using views. In *ACM Symp. on Principles of Database Systems (PODS)*, pages 95–104, 1995.

[9] Bohdan S Majewski and George Havas. The complexity of greatest common divisor computations. In *Algorithmic Number Theory*, pages 184–193. Springer, 1994.

[10] Alan Nash, Luc Segoufin, and Victor Vianu. Views and queries: Determinacy and rewriting. *ACM Transactions on Database Systems*, 35(3), 2010.