UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

# APROXIMACIONES EFICIENTES DE CONSULTAS CONJUNTIVAS

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN CIENCIAS, MENCIÓN COMPUTACIÓN
TESIS PARA OPTAR AL TÍTULO DE INGENIERO CIVIL MATEMÁTICO

MIGUEL ÁNGEL ROMERO ORTH

PROFESOR GUÍA:
PABLO BARCELÓ BAEZA

MIEMBROS DE LA COMISIÓN:
MARCOS KIWI KRAUSKOPF
GONZALO NAVARRO BADINO
JORGE PÉREZ ROJAS
MARCELO ARENAS SAAVEDRA

---

Este trabajo ha recibido el apoyo del Proyecto Fondecyt 1110171.

---

SANTIAGO DE CHILE
MAYO 2012

UNIVERSITY OF CHILE
FACULTY OF PHYSICS AND MATHEMATICS
DEPARTMENT OF COMPUTER SCIENCE

# EFFICIENT APPROXIMATIONS OF CONJUNCTIVE QUERIES

Submitted to the University of Chile in fulfillment of the thesis requirement to obtain the degree of MSc. in Computer Science and Engineering in Mathematics.

MIGUEL ÁNGEL ROMERO ORTH

ADVISOR:
PABLO BARCELÓ BAEZA

COMMITTE:
MARCOS KIWI KRAUSKOPF
GONZALO NAVARRO BADINO
JORGE PÉREZ ROJAS
MARCELO ARENAS SAAVEDRA

---

This work has been supported in part by Fondecyt Project 1110171.

---

SANTIAGO - CHILE
MAY 2012

*Dedicado a mi familia, a mis amigos*
*y a Daniela.*

# Resumen

Cuando encontrar la respuesta exacta a una consulta sobre una base de datos muy grande es intratable, es natural aproximar la consulta por otra más eficiente que pertenezca a una clase con buenas cotas en la complejidad de evaluación de consultas. En esta tesis estudiamos tales aproximaciones para consultas conjuntivas. Estas consultas son de especial interés en base de datos, y además sabemos muy bien qué clases de consultas admiten una evaluación eficiente, como las consultas acíclicas, o las de *(hyper)treewidth* acotado.

Definimos una aproximación a una consulta $Q$ como una consulta de una de esas clases que discrepa con $Q$ lo menos posible. Nos concentramos en aproximaciones que siempre entregan respuestas correctas. Probamos que para las clases tratables de consultas conjuntivas mencionadas anteriormente, siempre existen aproximaciones y sus tamaños son a lo más polinomiales en el tamaño de la consulta original. Esto se sigue de resultados generales obtenidos que relacionan propiedades de clausura de clases de consultas conjuntivas con la existencia de aproximaciones. Además, probamos que en muchos casos el tamaño de la aproximación es a lo más el tamaño de la consulta original. Presentamos una serie de resultados sobre cómo ciertas propiedades combinatoriales de las consultas afectan a sus aproximaciones y estudiamos cotas en la cantidad de aproximaciones, al igual que la complejidad de encontrar e identificar aproximaciones. Finalmente, consideramos aproximaciones que entregan todas las respuestas correctas y estudiamos sus propiedades.

# Abstract

When finding exact answers to a query over a large database is infeasible, it is natural to approximate the query by a more efficient one that comes from a class with good bounds on the complexity of query evaluation. In this thesis we study such approximations for conjunctive queries. These queries are of special importance in databases, and we have a very good understanding of the classes that admit fast query evaluation, such as acyclic, or bounded (hyper)treewidth queries.

We define approximations of a given query $Q$ as queries from one of those classes that disagree with $Q$ as little as possible. We concentrate on approximations that are guaranteed to return correct answers. We prove that for the above classes of tractable conjunctive queries, approximations always exist, and are at most polynomial in the size of the original query. This follows from general results we establish that relate closure properties of classes of conjunctive queries to the existence of approximations. We also show that in many cases, the size of approximations is bounded by the size of the query they approximate. We establish a number of results showing how combinatorial properties of queries affect properties of their approximations, study bounds on the number of approximations, as well as the complexity of finding and identifying approximations. We also look at approximations that return all correct answers and study their properties.

# Acknowledgements

I would like to thank my advisor, Pablo Barceló, whose guidance and immense support during this process was fundamental to develop this work. I also want to thank Leonid Libkin who gave me the opportunity to work with him in the Department of Computer Science of the University of Edinburgh. It was a rewarding experience.

This work has been supported in part by Fondecyt Project 1110171.

# Contents

# Chapter 1

# Introduction

The idea of finding approximate solutions to problems for which computing exact solutions is impossible or infeasible is ubiquitous in computer science. It is common in database research too: approximate query answering techniques are used for evaluating queries over extremely large databases or for queries with very high inherent complexity, see, e.g., [10, 11, 14, 23, 28]. By analyzing the structure of both the database and the query one can often find a reasonable approximation of the answer, sometimes with performance guarantees. Approximate techniques are relevant even for problems whose complexity is viewed as acceptable for regular-size databases, since finding precise answers may become impossible for large data sets we often deal with these days.

To approximate a query, we must have a good understanding of the complexity of query evaluation, in order to find an approximation that is guaranteed to be efficient. For one very common class of queries – *conjunctive*, or select-project-join queries – we do have a very good understanding of their complexity. In fact, we know which classes of conjunctive queries (CQs from now on) are easy to evaluate [8, 15, 16, 17, 24, 34]. Given the importance of conjunctive queries, and our good understanding of them, we would like to initiate a study of their approximations. We do it from the *static analysis* point of view, i.e., independently of the input database: for a query $Q$, we want to find another query $Q'$ that will be much faster than $Q$, and whose output would be close to the output of $Q$ on *all* databases. Such analysis is essential when a query is repeatedly evaluated on a very large database (say, in response to frequent updates), and when producing approximations based on both data and queries may be infeasible.

The complexity of checking whether a tuple $\bar{a}$ belongs to the output of a CQ $Q$ on a database $D$ is of the order $|D|^{O(|Q|)}$, where $|\cdot|$ measures the size (of a database or a query) [3, 33]. In fact, the problem is known to be NP-complete, when its input consists of $D$ as well as $Q$ (even for Boolean CQs). In other words, the *combined complexity* of CQs is intractable

[7]. Of course the *data complexity* of CQs is low, but having $O(|Q|)$ as the exponent may be prohibitively high for very large datasets. This observation led to an extensive study of classes of CQs for which the combined complexity is tractable. The first result of this kind by Yannakakis [34] showed tractability for *acyclic* CQs. That was later extended to queries of *bounded treewidth* [8, 12, 24]; this notion captures tractability for classes of CQs defined in terms of their graphs [17]. For classes of CQs defined in terms of their hypergraphs, the corresponding notion guaranteeing tractability is *bounded hypertree width* [16], which includes acyclicity as a special case. All these conditions can be tested in polynomial time [5, 13, 16].

The question we address is whether we can approximate a CQ $Q$ by a CQ $Q'$ from one of such classes so that $Q$ and $Q'$ would disagree as little as possible. Assume, for example, that we manage to find an approximation of $Q$ by an *acyclic* CQ $Q'$, for which checking whether $\bar{a} \in Q'(D)$ is done in time $O(|D| \cdot |Q'|)$ [34]. Then we replaced the original problem of complexity $|D|^{O(|Q|)}$ with that of complexity

$$O\big(f(|Q|) + |D| \cdot s(|Q|)\big)$$

where $s(\cdot)$ measures the size of the resulting approximation, and $f(\cdot)$ is the complexity of finding one.

Thus, assuming that the complexity measures $f$ and $s$ are acceptable, the combined complexity of running $Q'$ is much better than for $Q$. Hence, if the quality of the approximation $Q'$ is good too, then we may prefer to run the much faster query $Q'$ instead of $Q$, especially in the case of very large databases. Thus, we need to answer the following questions:

- What are the acceptable bounds for constructing approximations, i.e., the functions $f$ and $s$ above?
- What types of guarantees do we expect from approximations?

For the first question, if $Q'$ is of the same size as $Q$, or even if it polynomially increases the size, this is completely acceptable, as the exponent $O(|Q|)$ is now replaced by the factor $s(|Q|)$. For the complexity $f$ of static computation (i.e., transforming $Q$ to $Q'$), a single exponential is typically acceptable. Indeed, this is the norm in many static analysis and verification questions [29, 31], and of course $c^{|Q|}$ for a constant $c$ is significantly smaller than $|D|^{|Q|}$. Thus, in terms of their complexity, our desiderata for approximations are:

1. the approximating query should be at most polynomially larger than $Q$ – and ideally, bounded by the size of $Q$; and
2. the complexity of finding an approximating query should not exceed single-exponential.

As for the guarantees we expect from approximations, in general they can be formulated in two different ways. By doing it qualitatively we state that an approximation is a query that cannot be improved in terms of how much it disagrees with the query it approximates. Alternatively, to do it quantitatively, we define a measure of disagreement between two queries, and look for approximations whose measure of disagreement with the query they approximate is below a certain threshold.

Here we develop the qualitative approach to approximating CQs. For a given $Q$, we compare queries from some good (tractable) class $\mathcal{C}$ by how much they disagree with $Q$: to do so, we define an ordering $Q_1 \sqsubseteq_Q Q_2$ saying, intuitively, that $Q_2$ disagrees with $Q$ less often than $Q_1$ does. Then the best queries with respect to the ordering are our approximations from the class $\mathcal{C}$.

Furthermore, we require the approximations to return correct results. This approach is standard in databases (for instance, the standard approximation of query results in the settings of query answering using views and data integration is the notion of maximally contained rewriting [2, 19, 26]). However, we shall also briefly discuss what happens if we relax this requirement.

Our main goal is to explore approximations of arbitrary CQs by tractable CQs. We begin by studying queries on graphs (as the essential machinery needs to be developed for them), and then extend results to arbitrary databases. It turns out that approximations are guaranteed to exist for all the tractable classes of CQs mentioned earlier, which makes the notion worth studying.

In general, the structure of approximations will depend heavily on combinatorial properties of the (tableau of the) query $Q$ we try to approximate. Consider, for instance, a Boolean query $Q_1()$:–$E(x,y), E(y,z), E(z,x)$ over graphs. Its best acyclic approximation is $Q_1'()$:–$E(x,x)$, which is contained in every Boolean graph query and thus provides us with little information. It turns out that this will be the case whenever the tableau of the query is not a bipartite graph. Let $P_m(x_0,\ldots,x_m)$ be the CQ stating that $x_0,\ldots,x_m$ form a path of length $m$, i.e., $E(x_0,x_1),\ldots,E(x_{m-1},x_m)$. If we now look at

$$Q_2() \text{ :- } P_3(x,y,z,u), P_3(x',y',z',u'), E(x,z'), E(y,u')$$

(which has a cycle with variables $x,y,z',u'$), then it has a nontrivial acyclic approximation

$$Q_2'()\text{:–}P_4(x,y,z,u,v)$$

What changed is that the tableau of $Q_2$ is bipartite, which guarantees the existence of nontrivial approximations. This example provides a flavor of the results we establish. Of

| Class of queries | Type of approximation | Existence of approximation | Size of approximation | Time to compute approximation |
|---|---|---|---|---|
| Graph queries | Acyclic | always exists | at most $\|Q\|$ | single-exponential |
| | Treewidth $k$ | | | |
| Arbitrary queries | Acyclic | | polynomial in $\|Q\|$ | |
| | Hypertreewidth $k$ | | | |

Figure 1.1: Summary of results on approximations for conjunctive queries $Q$

course Boolean queries on graphs are just a useful test case, but they tell us for which classes of queries it makes sense to look for meaningful approximations.

## 1.1 Summary of results

We now provide a quick summary of the results of the thesis. As mentioned earlier, we first study queries over graphs and then lift results to arbitrary queries.

**Results for graph queries.** For a query $Q$, we are interested in approximations $Q'$ from a good class $\mathcal{C}$. The classes we consider are acyclic queries [34] and queries of fixed treewidth $k$, which capture the notion of tractability of CQs over graphs [17]. The first two rows in Figure 1.1 summarize some of our results: within both classes, approximations exist for all queries (this will follow from a general existence result that relates closure properties of classes of graphs to the existence of approximations), they do not increase the complexity of the query, and can be constructed in single-exponential time, thus satisfying all our desiderata for approximating queries.

We prove several additional results as well. We study the structure of approximations, and relate graph-theoretic properties of tableaux of queries with properties of their approximations (showing, for instance, a close relationship between $(k+1)$-colorability of the tableau and the existence of interesting treewidth-$k$ approximations). For Boolean queries, we show a finer trichotomy result for acyclic approximations, and also prove that such approximations are guaranteed to reduce the number of joins. As for the number of non-equivalent approximations, there are finitely many of them, in fact at most exponentially many in $|Q|$. We show that in fact, asymptotically, there can be at least exponentially many of them in the size of the query they approximate.

We provide further complexity analysis, showing that the problem of checking whether $Q'$ is an acyclic (or treewidth-$k$) approximation of $Q$ is complete for the class DP (this class, defined formally later, is "slightly" above both NP and coNP [30]). DP-completeness results appeared in the database literature in connection with computing cores of structures [9]; our

result is of a very different nature because it holds even when both $Q$ and $Q'$ are minimized (i.e., their tableaux are cores).

We also look at overapproximations which return all correct answers (and perhaps more). There the situation is quite different: acyclic overapproximations need not exist even for Boolean CQs. We provide some sufficient conditions for the non-existence of overapproximations, and show than when overapproximations of a query $Q$ do exist, they are unique (up to equivalence), and have strictly fewer joins than $Q$ itself.

**Results for arbitrary queries.** There are two ways of getting tractable classes of CQs over arbitrary databases, depending on whether one formulates conditions in terms of the *graph* of a query $Q$, or its *hypergraph*. For graph-based notions, it is known that bounded treewidth characterizes tractability [17]. For them, results for graph queries extend to arbitrary queries.

For hypergraph-based notions, we have the original notion of acyclicity from [34] and its more recent extension to the notion of bounded *hypertree width* [16]; it is known that hypertree width 1 coincides with acyclicity. We again prove a general existence result for approximations. However, the closure conditions imposed on classes of hypergraphs are becoming more involved, and it actually requires an effort to prove that they hold for classes of bounded hypertree width. We show that it is still possible to find approximations in single exponential time. As for their sizes, they need not be bounded by $|Q|$, but they remain polynomial in $|Q|$, with the polynomial depending only on the vocabulary (schema). Thus, as the summary table in Figure 1.1 shows, in this case too, our desiderata for approximations are met.

Regarding techniques required to prove these results, we mainly work with tableaux of queries, and characterize approximations via preorders based on the existence of homomorphisms. Thus, we make a heavy use of techniques from the theory of graph homomorphisms [20]. Besides graph theory and combinatorics, these are commonly used in constraint satisfaction [25], but recently they were applied in database theory as well [6, 27].

**Organization.** Basic notations are given in Chapter 2. In Chapter 3 we define the notion of approximations. Next, we study queries over graphs, concentrating on acyclic and bounded treewidth approximations (Chapter 4 and Chapter 5). We study the existence of approximations, as well as the number of them and their structure in Chapter 4. In Chapter 5 we concentrate in the complexity of identifying approximations. In Chapter 6 we discuss how to extend these results to arbitrary databases and we study overapproximations. Conclusions are given in Chapter 7.

**Note.** The results in this thesis are joint work with professors Pablo Barceló and Leonid Libkin. All the material in this thesis is from [36]. All the results were discussed by the three

authors. However, the proofs of the results that appear in the thesis were written by the author himself except for Proposition 4.3.6, Theorem 4.0.1 and Theorem 6.1.1.

# Chapter 2

# Preliminaries

## 2.1 Graphs and digraphs

Both graphs and digraphs are defined as pairs $G = \langle V, E\rangle$, where $V$ is a set of nodes (vertices) and $E$ is a set of edges. For graphs, an edge is a set $\{u, v\}$, where $u, v \in V$; for digraphs, an edge is a pair $(u, v)$, i.e., it has an orientation from $u$ to $v$. If $u = v$, we have a (undirected or directed) loop.

If $G = \langle V, E\rangle$ is a directed graph, then $G^{\mathrm{u}}$ is the underlying undirected graph: $G^{\mathrm{u}} = \langle V, \{\{u, v\} \mid \ (u, v) \in E\}\rangle$.

We denote by $K_m$ the complete graph on $m$ vertices: $K_m = \langle\{u_1, \ldots, u_m\}, \{\{u_i, u_j\} \mid i \neq j,\ i, j \leq m\}\rangle$, and by $K_m^{\rightleftarrows}$ the complete digraph on $m$ vertices, i.e., $K_m^{\rightleftarrows} = \langle\{u_1, \ldots, u_m\}, \{(u_i, u_j) \mid i \neq j,\ i, j \leq m\}$, so that edges go in both directions. Note that $(K_m^{\rightleftarrows})^{\mathrm{u}} = K_m$.

## 2.2 Graph homomorphisms and cores

Given two graphs (directed or undirected) $G_1 = \langle V_1, E_1\rangle$ and $G_2 = \langle V_2, E_2\rangle$, a *homomorphism* between them is a map $h : V_1 \to V_2$ such that $h(e)$ is in $E_2$ for every edge $e \in E_1$. Of course by $h(e)$ we mean $\{h(u), h(v)\}$ if $e = \{u, v\}$ and $(h(u), h(v))$ if $e = (u, v)$. The image of $h$ is the (di)graph $\mathrm{Im}(h) = \langle h(V_1), \{h(e) \mid e \in E_1\}\rangle$. If there is a homomorphism $h$ from $G_1$ to $G_2$, we write $G_1 \to G_2$ or $G_1 \xrightarrow{h} G_2$.

A graph $G$ is a *core* if there is no homomorphism $G \to G'$ into a proper subgraph $G'$ of $G$. A subgraph $G'$ of $G$ is a *core of* $G$ if $G'$ is a core and $G \to G'$. It is well known that all cores of a graph are isomorphic and hence we can speak of the core of a graph, denoted by $\mathrm{core}(G)$. We say that two graphs $G$ and $G'$ are *homomorphically equivalent* if both $G \to G'$ and $G' \to G$ hold. Homomorphically equivalent graphs have the same core, i.e., $\mathrm{core}(G)$ and $\mathrm{core}(G')$ are isomorphic.

We shall also deal with graphs with distinguished vertices. Let $G, G'$ be (di)graphs and $\bar{u}$, $\bar{u}'$ tuples of vertices in $G$ and $G'$, respectively, of the same length. Then we write $(G, \bar{u}) \to (G', \bar{u}')$ if there is a homomorphism $h : G \to G'$ such that $h(\bar{u}) = \bar{u}'$. With this definition, the notion of core naturally extends to graphs with distinguished vertices, considering that $(G', \bar{u}')$ is a subgraph of $(G, \bar{u})$ iff $G'$ is subgraph of $G$ and $\bar{u}' = \bar{u}$.

We write $G < G'$ if $G \to G'$, but $G' \to G$ does not hold.

## 2.3 Databases (relational structures)

While the case of graphs is crucial for understanding the main concepts, we shall also state results for conjunctive queries over arbitrary relational structures. A *vocabulary* (often called a *schema* in the database context) is a set $\sigma$ of relation names $R_1, \ldots, R_l$, each relation $R_i$ having an arity $n_i$. A *relational structure*, or a database, of vocabulary $\sigma$ is $\mathcal{D} = \langle U, R_1^{\mathcal{D}}, \ldots, R_l^{\mathcal{D}} \rangle$, where $U$ is a finite set, and each $R_i^{\mathcal{D}}$ is an $n_i$-ary relation over $U$, i.e., a subset of $U^{n_i}$. We usually omit the superscript $^{\mathcal{D}}$ if it clear from the context. We also assume (as is normal in database theory) that $U$ is the active domain of $\mathcal{D}$, i.e., the set of all elements that occur in relations $R_i^{\mathcal{D}}$'s.

Both directed and undirected graphs, for example, are relational structures of the vocabulary that contains a single binary relation $E$. For digraphs, it is the edge relation; for graphs, it contains pairs $(u, v)$ and $(v, u)$ for each edge $\{u, v\}$.

We often deal with databases together with a tuple of distinguished elements, i.e., $(\mathcal{D}, \bar{a})$, where $\bar{a}$ is a $k$-tuple of elements of the active domain, for some $k > 0$. Technically, these are structures of vocabulary $\sigma$ expanded with $k$ extra constant symbols, interpreted as $\bar{a}$.

Homomorphisms of structures are defined in the same way as for graphs: for $\mathcal{D}_1 = \langle U_1, (R_i^{\mathcal{D}_1})_{i \le l} \rangle$ and $\mathcal{D}_2 = \langle U_2, (R_i^{\mathcal{D}_2})_{i \le l} \rangle$, a homomorphism $h : \mathcal{D}_1 \to \mathcal{D}_2$ is a map from $U_1$ to $U_2$ so that $h(\bar{t}) \in R_i^{\mathcal{D}_2}$ for every $n_i$-ary tuple $\bar{t} \in R_i^{\mathcal{D}_1}$, for all $i \le l$. As before, we write $\mathcal{D}_1 \to \mathcal{D}_2$ in this case. For databases with tuples of distinguished elements we have $(\mathcal{D}_1, \bar{a}_1) \to (\mathcal{D}_2, \bar{a}_2)$ if the homomorphism $h$ in addition satisfies $h(\bar{a}_1) = \bar{a}_2$.

The notion of a core for relational structures (with distinguished elements) is defined just as for graphs, using homomorphisms of structures.

## 2.4 Conjunctive queries and tableaux

A conjunctive query (CQ) over a relational vocabulary $\sigma$ is a logical formula in the $\exists, \wedge$-fragment of first-order logic, i.e., a formula of the form $Q(\bar{x}) = \exists \bar{y} \bigwedge_{j=1}^{m} R_{i_j}(\bar{x}_{i_j})$, where each $R_{i_j}$ is a symbol from $\sigma$, and $\bar{x}_{i_j}$ a tuple of variables among $\bar{x}, \bar{y}$ whose length is the arity

of $R_{i_j}$. These are often written in a rule-based notation

$$Q(\bar{x}) :- R_{i_1}(\bar{x}_{i_1}), \ldots, R_{i_m}(\bar{x}_{i_m}). \tag{2.1}$$

The *number of joins* in the CQ (2.1) is $m-1$. Given a database $\mathcal{D}$, the answer $Q(\mathcal{D})$ to $Q$ is $\{\bar{a} \mid \mathcal{D} \models Q(\bar{a})\}$. If $Q$ is a Boolean query (a sentence), the answer *true* is, as usual, modeled by the set containing the empty tuple, and the answer *false* by the empty set.

A CQ $Q$ is *contained* in a CQ $Q'$, written as $Q \subseteq Q'$, if $Q(\mathcal{D}) \subseteq Q'(\mathcal{D})$ for every database $\mathcal{D}$.

With each CQ $Q(\bar{x})$ of the form (2.1) we associate its *tableau* $(T_Q, \bar{x})$, where $T_Q$ is the body of $Q$ viewed as a $\sigma$-database; i.e., it contains tuples $\bar{x}_{i_j}$'s in relations $R_{i_j}$'s, for $j \leq m$. If $Q$ is a Boolean CQ, then its tableau is just the $\sigma$-structure $T_Q$.

Many key properties of CQs can be stated in terms of homomorphisms of tableaux. For example, $\bar{a} \in Q(\mathcal{D})$ iff $(T_Q, \bar{x}) \to (\mathcal{D}, \bar{a})$. For CQs $Q(\bar{x})$ and $Q'(\bar{x}')$ with the same number of free variables, $Q \subseteq Q'$ iff $(T_{Q'}, \bar{x}') \to (T_Q, \bar{x})$. Hence, the combined complexity of CQ evaluation and the complexity of CQ containment are in NP (in fact, both are NP-complete [7]).

# Chapter 3

# The notion of approximation

We now explain the main idea of approximations. Suppose $\mathcal{C}$ is a class of conjunctive queries (e.g., acyclic, or of bounded treewidth). We are given a query $Q$ not in this class, and we want to approximate it within $\mathcal{C}$. For that, we define an ordering $\sqsubset_Q$ on queries in $\mathcal{C}$: the meaning of $Q_1 \sqsubset_Q Q_2$ is that "$Q_2$ approximates $Q$ better than $Q_1$ does", i.e., $Q_2$ agrees with $Q$ more often than $Q_1$. Then we look for maximal elements with respect to $\sqsubset_Q$ as good approximations of $Q$. As explained earlier, we typically are interested in queries that are guaranteed to return correct results.

We now formalize this. For queries $Q(\bar{x})$ and $Q'(\bar{x}')$, a database $\mathcal{D}$ and a tuple $\bar{a}$, we say that $Q$ *agrees with* $Q'$ on $(\mathcal{D}, \bar{a})$ if either $\bar{a}$ belongs to both $Q(\mathcal{D})$ and $Q'(\mathcal{D})$, or to none of them. Then, for CQs $Q(\bar{x}), Q_1(\bar{x}_1), Q_2(\bar{x}_2)$, with $\bar{x}, \bar{x}_1, \bar{x}_2$ of the same length, we define

$$Q_1 \sqsubseteq_Q Q_2 \stackrel{\text{def}}{=} \forall(\mathcal{D}, \bar{a}) \left( \begin{array}{c} Q_1 \text{ agrees with } Q \text{ on } (\mathcal{D}, \bar{a}) \\ \Downarrow \\ Q_2 \text{ agrees with } Q \text{ on } (\mathcal{D}, \bar{a}) \end{array} \right)$$

That is, $Q_2$ approximates $Q$ at least as well as $Q_1$ does. Then $Q_2$ approximates $Q$ better than $Q_1$ does if $Q_1 \sqsubset_Q Q_2$, i.e., $Q_1 \sqsubseteq_Q Q_2$ and $Q_2 \not\sqsubseteq_Q Q_1$.

**Definition 3.0.1. (Approximations)** *Given a class $\mathcal{C}$ of CQs and a query $Q$, a query $Q' \in \mathcal{C}$ such that $Q' \subseteq Q$ is a* $\mathcal{C}$-approximation *of $Q$ if there is no query $Q'' \in \mathcal{C}$ with $Q'' \subseteq Q$ such that $Q' \sqsubset_Q Q''$.*

In other words, $Q'$ is an approximation of $Q$ if it is guaranteed to return correct results and no other query approximates $Q$ better than $Q'$.

We next show that the use of the ordering $\sqsubseteq_Q$ in the definition can be replaced by a containment test:

**Proposition 3.0.2.** *Given a CQ $Q(\bar{x})$ and a CQ $Q'(\bar{x}') \in \mathcal{C}$ with the same number of free variables, the following are equivalent:*

1. *$Q'$ is a $\mathcal{C}$-approximation of $Q$;*
2. *$Q' \subseteq Q$, and there is no $Q'' \in \mathcal{C}$ such that $Q' \subset Q'' \subseteq Q$;*
3. *$(T_Q, \bar{x}) \to (T_{Q'}, \bar{x}')$, and there is no $Q''(\bar{x}'') \in \mathcal{C}$ such that $(T_Q, \bar{x}) \to (T_{Q''}, \bar{x}'') < (T_{Q'}, \bar{x}')$.*

To prove Proposition 3.0.2 we need the following claim:

**Claim 3.0.3.** *If $Q_1(\bar{x}_1)$ and $Q_2(\bar{x}_2)$ are CQs contained in CQ $Q(\bar{x})$, then $Q_1 \sqsubseteq_Q Q_2$ if and only if $(T_{Q_2}, \bar{x}_2) \to (T_{Q_1}, \bar{x}_1)$.*

**Proof.** Let $Q_1$ and $Q_2$ be contained in $Q$. Suppose $Q_1 \sqsubseteq_Q Q_2$. It holds that $Q_1$ agrees with $Q$ on $(T_{Q_1}, \bar{x}_1)$, since $(T_{Q_1}, \bar{x}_1) \to (T_{Q_1}, \bar{x}_1)$ and $(T_Q, \bar{x}) \to (T_{Q_1}, \bar{x}_1)$ $(Q_1 \subseteq Q)$. Therefore, $Q_2$ agrees with $Q$ on $(T_{Q_1}, \bar{x}_1)$, implying that $(T_{Q_2}, \bar{x}_2) \to (T_{Q_1}, \bar{x}_1)$. On the other hand, suppose that $(T_{Q_2}, \bar{x}_2) \to (T_{Q_1}, \bar{x}_1)$. We shall prove that $Q_1 \sqsubseteq_Q Q_2$. Let $(D, \bar{a})$ be a database with distinguished elements such that $Q_1$ and $Q$ agree on it. Suppose first, that $(T_{Q_1}, \bar{x}_1) \to (D, \bar{a})$ and $(T_Q, \bar{x}) \to (D, \bar{a})$. Then, by composition it holds that $(T_{Q_2}, \bar{x}_2) \to (D, \bar{a})$, implying that $Q_2$ and $Q$ agree on $(D, \bar{a})$. Now suppose that $(T_{Q_1}, \bar{x}_1) \not\to (D, \bar{a})$ and $(T_Q, \bar{x}) \not\to (D, \bar{a})$. Necessarily, $(T_{Q_2}, \bar{x}_2) \not\to (D, \bar{a})$, otherwise, since $(T_Q, \bar{x}) \to (T_{Q_2}, \bar{x}_2)$ $(Q_2 \subseteq Q)$, we would have $(T_Q, \bar{x}) \to (D, \bar{a})$. Therefore, $Q_2$ and $Q$ agree on $(D, \bar{a})$ in this case too, concluding that $Q_1 \sqsubseteq_Q Q_2$. □

**Proof of Proposition 3.0.2.** ((1) ⇒ (2)): By contradiction, suppose that there exists $Q'' \in \mathcal{C}$ such that $Q' \subset Q'' \subseteq Q$. Since $Q'$ and $Q''$ are contained in $Q$, and $Q' \subseteq Q''$ and $Q'' \not\subseteq Q'$, using Claim 3.0.3, it follows that $Q' \sqsubseteq_Q Q''$ and $Q'' \not\sqsubseteq_Q Q'$, that is, $Q' \sqsubset_Q Q''$, which contradicts (1).

((2) ⇒ (3)): Trivial.

((3) ⇒ (1)): By contradiction, suppose that there exists $Q'' \in \mathcal{C}$ such that $Q'' \subseteq Q$ and $Q' \sqsubset_Q Q''$. Again, using Claim 3.0.3, we have that $(T_Q, \bar{x}) \to (T_{Q''}, \bar{x}'') < (T_{Q'}, \bar{x}')$, which contradicts (3). □

Observe that Proposition 3.0.2 is very useful, as now we can use all the homomorphism theory machinery to analyze approximations.

Before describing the classes in which we shall try to approximate CQs, we present a useful view of approximations via orderings on queries and tableaux.

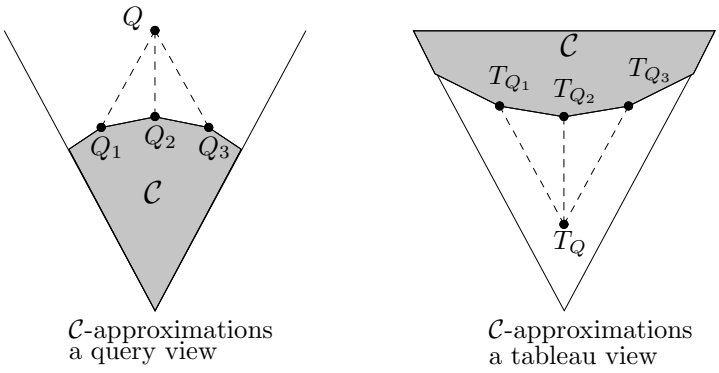


Figure 3.1: $\mathcal{C}$-approximations: an illustration

## 3.1 Approximations via ordering

Both CQs and their tableaux come naturally equipped with two preorders: containment of CQs, and the existence of homomorphisms between tableaux. These preorders are dual to each other [7]: $Q \subseteq Q' \Leftrightarrow T_{Q'} \to T_Q$. These relations are reflexive and transitive but not antisymmetric (as we may have different equivalent queries), hence they are preorders. They become partial orders when restricted to cores, or minimized CQs. Indeed, if both $T_{Q'} \to T_Q$ and $T_Q \to T_{Q'}$ hold, then $T_{Q'}$ and $T_Q$ are homomorphically equivalent and thus have the same core (which happens to be the tableau of the minimized version of $Q$). The preorder $\to$ and its restriction to cores have been actively studied over graphs, digraphs, and relational structures [20], and we shall heavily use their properties in our proofs.

With this view, we can visualize the result of Proposition 3.0.2 as shown in Fig. 3.1. The $\mathcal{C}$-approximations of $Q$ are the "closest" elements of class $\mathcal{C}$ that are below $Q$ in the $\subseteq$ ordering. If we switch to the tableau view, then approximations are the closest elements of $\mathcal{C}$ which are above the tableau of $Q$ in the $\to$ ordering.

## 3.2 Good classes of queries

We look for approximations within tractable classes of CQs, which include acyclic queries, as well as queries of bounded treewidth and hypertree width [8, 12, 16, 17, 24, 34]. We now define the first two (hypertree width is defined in Chapter 6).

We first need the notion of tree decompositions of hypergraphs of queries. Recall that a hypergraph $\mathcal{H} = \langle V, \mathcal{E} \rangle$ has a set of vertices $V$ and a set of hyperedges $\mathcal{E}$; each hyperedge is a subset of $V$. For a CQ $Q$, its hypergraph $\mathcal{H}(Q)$ has all the variables used in $Q$ as vertices; the hyperedges are sets of variables that appear in the same atom. For example, for the query

with the body $R(x, y, z), R(x, v, v), E(v, z)$, the hyperedges are $\{x, y, z\}$, $\{x, v\}$, and $\{v, z\}$.

A *tree decomposition* of a hypergraph $\mathcal{H} = \langle V, \mathcal{E} \rangle$ is a tree $T$ together with a map $f : T \to 2^V$ that associates a set of vertices in $V$ with each node of $T$ such that

1. each hyperedge from $\mathcal{E}$ is contained in one of the sets $f(u)$ for $u \in T$; and
2. for every $v \in V$, the set $\{u \in T \mid v \in f(u)\}$ is a connected subset of $T$.

The *width* of a decomposition is $\max_{u \in T} |f(u)| - 1$, and the *treewidth* of $\mathcal{H}$ is the minimum width of its tree decompositions. If $\mathcal{H}$ is a tree (or a forest) to start with, then its treewidth is 1. We refer to the classes of hypergraphs of treewidth at most $k$ as $\mathsf{TW}(k)$, and, slightly abusing notation, we use $\mathsf{TW}(k)$ to also denote the classes of CQs (and their tableaux) whose hypergraphs have treewidth at most $k$.

A hypergraph is *acyclic* if there is a tree decomposition $(T, f)$ of it such that every $f(u)$ is a hyperedge. A CQ is acyclic if its hypergraph is acyclic. We use $\mathsf{AC}$ to denote the class of acyclic hypergraphs (and also acyclic CQs, and their tableaux). For queries over graphs, we have $\mathsf{AC} = \mathsf{TW}(1)$. In general the notions of bounded treewidth and acyclicity are incompatible (see, e.g., [12]).

**Remark** The notion of approximation is based on the semantics of queries, while the good classes are defined purely by syntactic means. It may happen that a query $Q$ not from a good class $\mathcal{C}$ is equivalent to some query $Q'$ from $\mathcal{C}$. In that case, $\mathcal{C}$-approximations of $Q$ are simply equivalent to $Q$.

# Chapter 4

# Existence of approximations

We are interested in approximations of arbitrary queries, i.e., with an arbitrary vocabulary $\sigma$. However, in this chapter and in the following one we concentrate on queries over graphs as the fundamental problems and techniques are already present in this setting. Nevertheless, in Chapter 6 we briefly discuss how to extend some results to arbitrary queries.

Thus, our vocabulary $\sigma$ has a single binary relation $E(\cdot,\cdot)$, interpreted as a directed graph. Given a CQ $Q(\bar{x})$, its tableau $(T_Q, \bar{x})$ is a digraph as well, with a distinguished tuple of elements $\bar{x}$. Thus, we shall define classes of queries in terms of classes $\mathcal{C}$ of graphs: a CQ $Q$ is a *$\mathcal{C}$-query* iff the graph $T_Q$ is in $\mathcal{C}$.

The standard tractable classes of acyclic CQs and treewidth-$k$ CQs do arise in this way (we shall explain this shortly). But first we prove a very general result on the existence of approximations, which shows good behavior of those for many classes of queries.

**Theorem 4.0.1.** *Let $\mathcal{C}$ be a class of graphs closed under taking subgraphs. Then every CQ $Q$ that has at least one $\mathcal{C}$-query contained in it also has a $\mathcal{C}$-approximation.*

*Moreover, the number of non-equivalent $\mathcal{C}$-approximations of $Q$ is at most exponential in the size of $Q$, and every $\mathcal{C}$-approximation of $Q$ is equivalent to one which has at most as many joins as $Q$.*

**Proof.** Given $Q(\bar{x})$ which is not a $\mathcal{C}$-query, let $H^{\mathcal{C}}(Q)$ be the set of all $\mathcal{C}$-queries whose tableaux are of the form $(\mathrm{Im}(h), h(\bar{x}))$, where $h$ is a homomorphism defined on $(T_Q, \bar{x})$. All such queries are contained in $Q$. Up to equivalence (renaming of variables) there are finitely many elements in $H^{\mathcal{C}}(Q)$. Moreover, it is nonempty. Indeed, there is a $\mathcal{C}$-query $Q'(\bar{x}')$ with $Q' \subseteq Q$ and hence $(T_Q, \bar{x}) \xrightarrow{h} (T'_Q, \bar{x}')$ for some $h$ (thus $h(\bar{x}) = \bar{x}'$). By the closure under subgraphs we know that $(\mathrm{Im}(h), \bar{x}')$ is a tableau of a $\mathcal{C}$-query. Now consider minimal elements, with respect to the preorder $\to$, in the set $H^{\mathcal{C}}(Q)$. We claim that they are $\mathcal{C}$-approximations of $Q$. Indeed let $(\mathrm{Im}(h_0), \bar{x}')$ be one such element, with $\bar{x}' = h_0(\bar{x})$. If it

is not a $\mathcal{C}$-approximation, then we have $(T_Q, \bar{x}) \xrightarrow{g} (T, \bar{x}'') \xrightarrow{g_1} (\mathrm{Im}(h_0), \bar{x}')$ for some homomorphisms $g$ and $g_1$ such that $(\mathrm{Im}(h_0), \bar{x}') \not\to (T, \bar{x}'')$, with $T \in \mathcal{C}$. Hence we have $(T_Q, \bar{x}) \xrightarrow{g} (\mathrm{Im}(g), \bar{x}'') \xrightarrow{g_1} (\mathrm{Im}(h_0), \bar{x}')$, as well as $(\mathrm{Im}(h_0), \bar{x}') \not\to (\mathrm{Im}(g), \bar{x}'')$, and $\mathrm{Im}(g)$ is in $\mathcal{C}$ since $\mathcal{C}$ is closed under taking subgraphs. Hence $(\mathrm{Im}(g), \bar{x}'')$ is in $H^{\mathcal{C}}(Q)$, and by the minimality of $(\mathrm{Im}(h_0), \bar{x}')$ we have a contradiction.

If $Q'(\bar{x}')$ is a $\mathcal{C}$-approximation, then $(T_Q, \bar{x}) \xrightarrow{h} (T_{Q'}, \bar{x}')$ and thus $(T_Q, \bar{x}) \xrightarrow{h} (\mathrm{Im}(h), \bar{x}')$, with $\mathrm{Im}(h)$ being a subgraph of $T_{Q'}$, and thus in $\mathcal{C}$. By Proposition 3.0.2 this implies that $(\mathrm{Im}(h), \bar{x}')$ and $(T_{Q'}, \bar{x}')$ are homomorphically equivalent, and $(\mathrm{Im}(h), \bar{x}')$ is a $\mathcal{C}$-approximation equivalent to $Q'$. Hence, all $\mathcal{C}$-approximations can be chosen to be of the form $(\mathrm{Im}(h), \bar{x}')$, which shows that there are at most exponentially many of them, and that they need not have more joins than $Q$. □

## 4.1 Acyclic and treewidth-$k$ approximations

We now explain how acyclic graph queries and treewidth-$k$ graph queries appear as $\mathcal{C}$-queries for appropriately chosen classes $\mathcal{C}$. An undirected graph can be viewed as a hypergraph (in which all hyperedges are of cardinality 1 or 2), and thus a graph query $Q$ is acyclic, or of treewidth-$k$, if $T_Q^{\mathrm{u}}$, the underlying graph of its tableau, is acyclic (as a hypergraph), or has treewidth at most $k$. We still refer to these as $\mathsf{AC}$ and $\mathsf{TW}(k)$. Notice that acyclicity is not the graph acyclicity since loops are allowed: for instance, the undirected graph with an edge between nodes $x$ and $y$ and a loop on $x$ is acyclic, since the hypergraph with hyperedges $\{x, y\}$ and $\{x\}$ is acyclic. Also we formulate these conditions in terms of undirected graphs: for instance, the query $Q() \text{:-} E(x, y), E(y, x)$ is acyclic, since $T_Q^{\mathrm{u}}$ contains a single edge between $x$ and $y$. Basically, acyclicity disallows cycles of length 3 or more.

There is a *trivial* query that belongs to $\mathsf{AC}$ and all $\mathsf{TW}(k)$'s that every other CQ $Q$ contains. Indeed, let $K_1^{\circlearrowleft}$ be a single-element loop, i.e., a graph with a single node $x$ and a loop $(x, x)$ on that node. Then, for each query $Q(\bar{x})$ with $m$ free variables, we have, via a constant homomorphism: $(T_Q, \bar{x}) \to (K_1^{\circlearrowleft}, (x, \ldots, x))$, and thus $Q'(x, \ldots, x) \text{ :- } E(x, x)$ is contained in $Q$.

This, together with the closure of $\mathsf{AC}$ and $\mathsf{TW}(k)$ under taking subgraphs, gives us:

**Corollary 4.1.1.** *Every CQ $Q$ over graphs has an acyclic approximation, as well as a treewidth-$k$ approximation, for each $k > 0$.*

## 4.2 Size and number of approximations

Let $\mathcal{C}$-APPR$(Q)$ be the set of all $\mathcal{C}$-approximations of $Q$. For example, AC-APPR$(Q)$ is the class of acyclic approximations of $Q$. These sets are nonempty when $\mathcal{C}$ is AC or TW$(k)$. They are infinite, but for a simple reason: each CQ has infinitely many equivalent CQs.

It is well known though [7] that each CQ $Q(\bar{x})$ has a unique (up to renaming of variables) equivalent minimal query: in fact, this is the query whose tableau is core$(T_Q, \bar{x})$. It is obtained by the standard process of minimization of CQs. We thus denote by $\mathcal{C}$-APPR$_{\min}(Q)$ the set of all minimizations of $\mathcal{C}$-approximations of $Q$.

From Corollary 4.1.1 and Theorem 4.0.1 we obtain:

**Corollary 4.2.1.** *For every CQ $Q$, both* AC-APPR$_{\min}(Q)$ *and* TW$(k)$-APPR$_{\min}(Q)$ *are finite nonempty sets of queries. The number of queries in those sets is at most exponential in the size of $Q$, and each one has at most as many joins as $Q$. Moreover, a query from* AC-APPR$_{\min}(Q)$ *or* TW$(k)$-APPR$_{\min}(Q)$ *can be constructed in single-exponential time in $|Q|$.*

Hence, acyclic and treewidth-$k$ approximations fulfill the criteria from the introduction: they always exist, they are not more complex than the original query, and they can be found with reasonable complexity.

As for the complexity of finding an approximation, one can easily see that the algorithm that simply checks homomorphisms on $T_Q$ and selects one whose image is minimal with respect to $\rightarrow$ runs in time $2^{O(n \cdot \log n)}$, where $n$ is the number of variables in $Q$. We shall discuss the complexity in more detail in Chapter 5.

As for the number of elements of $\mathcal{C}$-APPR$_{\min}(Q)$, a simple upper bound is $2^{n \cdot \log n}$ (a better bound is the $n$th Bell number [4]). This raises the question whether the exponential number of approximating queries can be witnessed. We prove that this is the case.

**Proposition 4.2.2.** *There is a family $(Q_n)_{n>0}$ of Boolean CQs over graphs such that the number of variables and joins in $Q_n$'s grows linearly with $n$, and $|$*AC-APPR$_{\min}(Q_n)| \geq 2^n$ *for all $n > 0$.*

In order to prove Proposition 4.2.2 we need some definitions and lemmas, from [20]. An *oriented path* $P = [u_0, ..., u_n]$ is a digraph with vertices $u_0, \ldots, u_n$ and $n$ edges such that either $(u_i, u_{i+1})$ or $(u_{i+1}, u_i)$ is an edge, for each $0 \leq i < n$. The vertex $u_0$ is the *initial vertex* of the oriented path and the vertex $u_n$ the *terminal vertex*. The *length* of $P$, denoted by $|P|$, is $n$. Typically, we depict an oriented path $P$ as an edge $uv$ labeled with $P$, representing that the initial vertex of $P$ is $u$ and the terminal vertex is $v$.

We define the *net length* of $P$, denoted by $nl(P)$, to be the number of forward edges minus the number of backward edges of $P$. Without loss of generality, we assume that the net length is always non negative (choosing the right enumeration of the oriented path).

Usually, we will write oriented paths or oriented cycles (see definition in Section 4.3) as strings in $\{0,1\}^*$, where 0 represents a forward edge and 1 represents a backward edge. For example, $P = 001$ means that $P$ is the oriented path with two forward edges followed by a backward edge.

For a balanced digraph $G$ (see definition in Section 4.3) and a vertex $v$ in $G$, we define the *level* of $v$ to be

$$l_G(v) = max\{nl(P) : P \text{ is an oriented path in } G \text{ with terminal vertex } v\}$$

Observe that $l_G(v)$ is finite since $G$ is balanced. We define the *height* $hg(G)$ of $G$ to be

$$hg(G) = max\{l_G(v) : v \in V(G)\}$$

There is an algorithmic way to think about levels. Consider a balanced digraph $G$ and the following algorithm: In each component of $G$, pick a vertex and label it 0. Once a vertex has been labeled by $i$, label all of its outneighbors (neighbors with an edge from the vertex to them) by $i+1$ and all of its inneighbors (neighbors with an edge from them to the vertex) by $i-1$. It can be verified, since $G$ is balanced, that this labeling produces unique labels (after the initial choices of one vertex per component with label 0). Finally, relabel all vertex, so that the smallest label is 0. It can be proved that the labels produced by the previous algorithm are precisely the levels of $G$. We have the following lemmas, from [20]:

**Lemma 4.2.3.** *If $G$ and $H$ are two balanced digraphs such that $G \to H$, then $hg(G) \leq hg(H)$.*

**Lemma 4.2.4.** *Let $G$ and $H$ be two balanced digraphs of the same height, then any homomorphism from $G$ into $H$ preserves the levels of vertices.*

Now we prove Proposition 4.2.2. Let $P$ and $P'$ be oriented paths. We define the digraph $D(P,P')$ as follows: Consider the digraph $(\{a,b,c,d\},\{(a,b),(a,d),(c,b),(c,d)\})$. Add disjoint copies of $P$ and $P'$ and identify the initial vertex of the copy of $P$ and $P'$, with $b$ and $d$, respectively. Finally, add disjoint copies of $P$ and $P'$ again, and identify the terminal vertex of the copy of $P$ and $P'$, with $a$ and $c$, respectively. See Figure 4.1.

Now, for oriented paths $P$ and $P'$, we define $D_{ac}(P,P')$ and $D_{bd}(P,P')$ as the digraphs obtained from $D(P,P')$ by identifying $a$ with $c$, and $b$ with $d$, respectively. See Figure 4.2.

We have the following claim:

**Claim 4.2.5.** *Let $P$ and $P'$ be incomparable ($P \not\to P'$ and $P' \not\to P$) oriented paths of the same net length $k > 0$, such that each interior vertex (vertex different from the initial and*

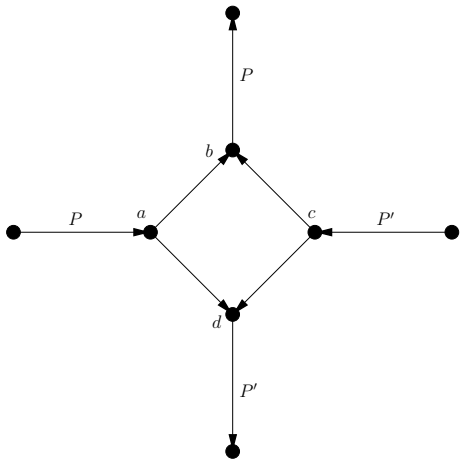


Figure 4.1: The digraph $D(P, P')$.

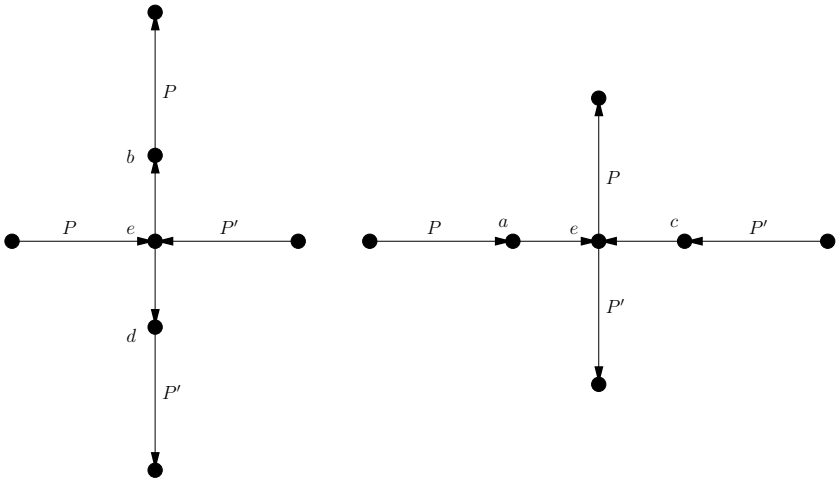


Figure 4.2: The digraphs $D_{ac}(P, P')$ and $D_{bd}(P, P')$.

*terminal vertices) in $P$ and $P'$ has a level that is neither 0 nor $k$. Then $D_{ac}(P, P')$ and $D_{bd}(P, P')$ are incomparable cores.*

**Proof.** Suppose that $D_{ac}(P, P')$ is not a core. Then $D_{ac}(P, P') \xrightarrow{h} D_{ac}(P, P')$, where $h$ is not surjective. Using Lemma 4.2.4, we know that $h$ preserves levels. It follows that $h(e) = e$ (see Figure 4.3). Now, $h(x_1)$ is either $x_1$ or $x_3$. Note that $h(x_1) = x_3$, implies that $P \to P'$, since no vertex in the copy of $P$ between $x_1$ and $e$ can be mapped to $b$ or $d$, and no vertex, except for the terminal one, has level $k$. It follows that $h(x_1) = x_1$. Similarly, we have that $h(x_3) = x_3$. Using the same argument, we have that $h(b) = b$, otherwise $h(b) = d$ and $P \to P'$, since no vertex in the copy of $P$ between $b$ and $x_2$ can be mapped to $e$. Similarly, $h(d) = d$. It follows that $h$ is surjective, which is a contradiction. Analogously, we have that

$D_{bd}(P, P')$ is a core.

Suppose that $D_{ac}(P, P') \xrightarrow{h} D_{bd}(P, P')$. Since $nl(P) = nl(P')$, $D_{ac}(P, P')$ and $D_{bd}(P, P')$ have height $2k+1$ (see, again, Figure 4.3). Using Lemma 4.2.4, we have that $h$ preserves levels. It follows that $h(e)$ is either $a$ or $c$. If $h(e) = a$, since no vertex in the copy of $P'$ between $x_3$ and $e$ can be mapped to $e$ in $D_{bd}(P, P')$ , it follows that $P' \to P$. Similarly, if $h(e) = c$, it follows that $P \to P'$. In any case, we have a contradiction. The case $D_{bd}(P, P') \not\to D_{ac}(P, P')$ is similar. □

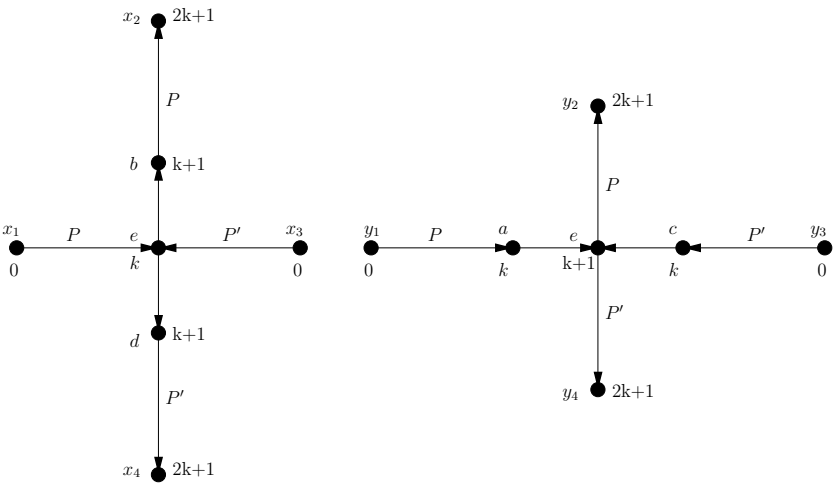


Figure 4.3: The levels of $D_{ac}(P, P')$ and $D_{bd}(P, P')$ when $nl(P) = nl(P') = k > 0$.

Consider the oriented path $P_1 = 001000$ and $P_2 = 000100$. We can easily show, using Lemma 4.2.4, that $P_1$ and $P_2$ are incomparable cores. For $n \geq 1$, we define $G_n$ to be the digraph constructed as follows: Take the union of $n$ disjoint copies of the digraph $D(P_1, P_2)$. For each $1 \leq i < n$, add an edge from the terminal vertex of the copy of $P_2$ in $i$-th copy of $D(P_1, P_2)$ which starts in $d$ to the initial vertex of the copy of $P_1$ in the $(i + 1)$-th copy of $D(P_1, P_2)$ which ends in $a$. The digraph $G_n$ is shown in Figure 4.4.

Consider a string $s \in \{V, H\}^n$ for $n \geq 1$. We define $G_n^s$ to be the digraph obtained from $G_n$ by identifying in the $i$-th copy of $D(P_1, P_2)$, for each $1 \leq i \leq n$, $a$ with $c$ if $s_i = V$ or $b$ with $d$ if $s_i = H$. We have the following claim:

**Claim 4.2.6.** *Let $s, s' \in \{V, H\}^n$ for $n \geq 1$, such that $s \neq s'$. Then $G_n^s$ and $G_n^{s'}$ are incomparable cores.*

**Proof.** Suppose $G_n^s$ is not a core. Then $G_n^s \xrightarrow{h} G_n^s$, where $h$ is not surjective. Observe that all the different copies of $D(P_1, P_2)$ with the vertices identified according to the string $s$, have disjoint levels between them, as shown in Figure 4.4. It follows that $h$ maps the $i$-th copy

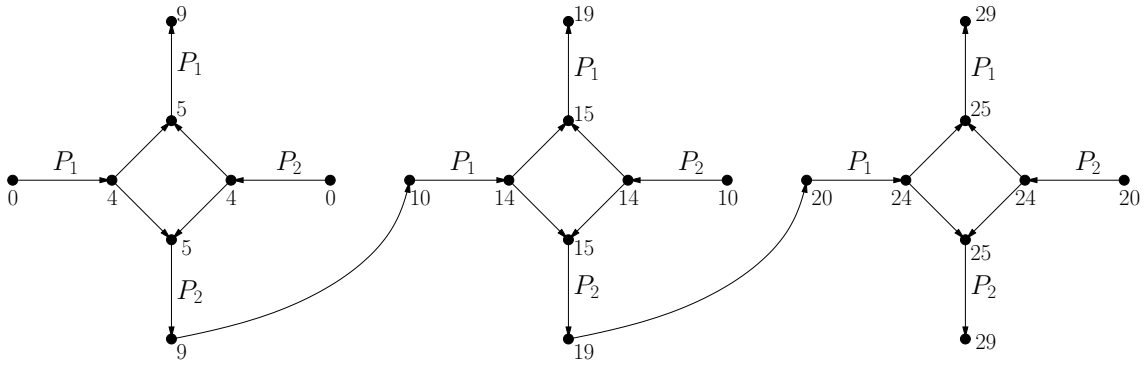


Figure 4.4: The digraph $G_3$ and some of its levels.

of $D(P_1, P_2)$ with the vertices identified to the same copy. Since $D_{ac}(P_1, P_2)$ and $D_{bd}(P_1, P_2)$ are cores (Claim 4.2.5), it follows that $h$ is surjective.

Now, assume that $G_n^s \xrightarrow{h} G_n^{s'}$. Since $G_n^s$ and $G_n^{s'}$ have the same height, we have that $h$ preserves levels. Again, we have that $h$ maps the $i$-th copy of $D(P_1, P_2)$ with the vertices identified in $G_n^s$ to the $i$-th copy of $D(P_1, P_2)$ with the vertices identified in $G_n^{s'}$. Since there exists $1 \leq j \leq n$ such that $s_j \neq s'_j$, we have that $D_{ac}(P_1, P_2)$ and $D_{bd}(P_1, P_2)$ are not incomparable, which is a contradiction with Claim 4.2.5. □

Now, we state two simple claims about $\sigma$-structures that will be useful in this proof and in other proofs as well:

**Claim 4.2.7.** *If $A$ is a $\sigma$-structure and $A'$ is the $\sigma$-structure obtained from $A$ by identifying two elements $a$ and $b$ of $A$ with a new element $z$ (that is, $A'$ is obtained from $A$ by replacing $a$ and $b$ by $z$ everywhere), then $A \to A'$.*

**Proof.** Just map $a$ and $b$ to $z$ and all the other elements to themselves. □

**Claim 4.2.8.** *Let $A$ and $B$ be two $\sigma$-structures. If $A \xrightarrow{h} B$ and $h(a) = h(b)$ for $a$ and $b$ in $A$, then $A' \to B$, where $A'$ is the $\sigma$-structure obtained from $A$ by identifying $a$ and $b$ in a new element $z$.*

**Proof.** Just map $z$ to $h(a) = h(b)$ and all the other elements to themselves. □

Now, we shall conclude the proof of the proposition. For each $n \geq 1$ and $s \in \{V, H\}^n$, let $Q_n$ be the CQ whose tableau is $G_n$ and $Q_n^s$ the CQ whose tableau is $G_n^s$. Observe that $Q_n$ is a cyclic CQ and $Q_n^s$ is an acyclic CQ. We have the next claim:

**Claim 4.2.9.** *$Q_n^s$ is an acyclic approximation of $Q_n$, for each $n \geq 1$ and $s \in \{V, H\}^n$.*

**Proof.** By contradiction, suppose that there exists an acyclic CQ $Q''$ such that $G_n \xrightarrow{h} T_{Q''} < G_n^s$. Consider the $i$-th copy of $D(P_1, P_2)$ in $G_n$. Necessarily, if we restrict $h$ to such copy, it holds that $h(a) = h(c)$ or $h(b) = h(d)$. Otherwise $h(a)$, $h(b)$, $h(c)$ and $h(d)$ would form an oriented cycle in $T_{Q''}$. If $h(a) = h(c)$, define $t_i = V$, otherwise define $t_i = H$. Now, define $t = t_1 t_2 ... t_n$. Using Claim 4.2.8, we have that $G_n^t \to T_{Q''}$ and by composition, it follows that $G_n^t \to G_n^s$. Using Claim 4.2.6, we have that $t = s$ and then $G_n^s \to T_{Q''}$, which is a contradiction. □

Observe that for each $n \geq 1$ and $s \in \{V, H\}^n$, $Q_n^s \in \mathsf{AC\text{-}APPR}_{\min}(Q_n)$. Therefore, for all $n \geq 1$,

$$|\mathsf{AC\text{-}APPR}_{\min}(Q_n)| \geq 2^n$$

Furthermore, observe that the number of variables in $Q_n$ (denoted by $|Q_n|$) is $28n$ and the number of joins are the numbers of edges in $G_n$ minus 1, that is, $29n - 2$. If follows that the family $(Q_n)_{n\geq 1}$ satisfies the required conditions.

## 4.3 Acyclic approximations

We now study acyclic approximations in more detail. We begin with the case of Boolean queries, when the tableau of a query is just a graph, and show a trichotomy theorem for them, classifying approximations based on graph-theoretic properties of the tableau. Then we extend results to arbitrary queries. After that we study the complexity of acyclic approximations.

### 4.3.1 Boolean queries

These queries are of the form $Q()$:–... and thus produce yes/no answers; their tableaux are simply directed graphs $T_Q$. We already talked about them in the introduction, and mentioned that for nontrivial approximations, the tableau must be *bipartite*. Recall that a digraph $G$ is bipartite if $G \to K_2^{\rightleftarrows}$, i.e., $G$ is 2-colorable: its nodes can be split into two disjoint subsets $A$ and $B$ so that all edges have endpoints in different subsets.

Recall the example from the introduction: the cyclic query $Q_1()$:–$E(x, y), E(y, z), E(z, x)$ had a *trivial* acyclic approximation $Q^{\text{triv}}()$:–$E(x, x)$ (which is contained in every Boolean graph query). The reason for that was $T_{Q_1}$ was not bipartite. In the introduction, we saw an example of a query with a bipartite tableaux that had a nontrivial approximation stating the existence of a path of length 4. Note that every query whose tableau is bipartite will contain the *trivial bipartite* query $Q_2^{\text{triv}}()$:–$E(x, y), E(y, x)$, whose tableau is $K_2^{\rightleftarrows}$. For some

cyclic queries, e.g.,

$$Q_3() \text{:-} E(x,y), E(y,z), E(z,u), E(x,u).$$

this trivial query is the only acyclic approximation. This behavior is caused by the cycle being unbalanced. We next define this concept [20], and then state the trichotomy result.

An *oriented cycle* is a digraph with vertices $u_1, \ldots, u_n$ and $n$ edges such that either $(u_i, u_{i+1})$ or $(u_{i+1}, u_i)$ is an edge, for each $i < n$, and either $(u_1, u_n)$ or $(u_n, u_1)$ is an edge. We shall refer to edges $(u_i, u_{i+1})$ and $(u_n, u_1)$ as *forward edges* and to edges $(u_{i+1}, u_i)$ and $(u_1, u_n)$ as *backward edges*. An oriented cycle is *balanced* if the number of forward edges equals the number of backward edges, and a digraph is balanced if every oriented cycle in it is balanced.

**Theorem 4.3.1.** *Let $Q$ be a Boolean CQ over graphs. Then, if its tableau $T_Q$:*

- *is not bipartite, then $Q$ has only the trivial acyclic approximation $Q^{\text{triv}}$;*
- *is bipartite but not balanced, then $Q$'s only acyclic approximation is the trivial bipartite query $Q_2^{\text{triv}}$;*
- *is bipartite and balanced, then none of $Q$'s acyclic approximations is trivial, and none contains two subgoals of the form $E(x,y), E(y,x)$.*

Of course when we talk about the only approximation, we mean it up to query equivalence. Note that the conditions used in the theorem – being bipartite and balanced – can be checked in polynomial time [20, 35].

First, we need the following claim:

**Claim 4.3.2.** *Balanced digraphs are closed under homomorphisms. That is, if $G \to H$ and $H$ is balanced, then $G$ is balanced.*

**Proof.** We use the following characterization of balanced digraphs from [20]: $G$ is balanced if and only if $G \to \vec{P}_k$, for some $k \geq 1$, where $\vec{P}_k$ denotes the directed path of length $k$. Now, suppose $G \to H$ and $H$ is balanced. Then $H$ is homomorphic to a directed path, and, by composition, $G$ is homomorphic to such directed path as well. Therefore, $G$ is balanced. □

**Proof of Theorem 4.3.1.** Suppose the tableau $T_Q$ of $Q$ is not bipartite and let $Q'$ be an acyclic approximation of $Q$. Necessarily, $T_{Q'}$ has a loop implying that $Q'$ is equivalent to $Q^{\text{triv}}$. Indeed, if $T_{Q'}$ has no loops then it is bipartite (since $Q'$ is acyclic). Since $T_Q \to T_{Q'}$ and $T_{Q'} \to K_2^{\rightleftarrows}$, it follows that $T_Q$ is bipartite, which is a contradiction.

Now, suppose that $T_Q$ is bipartite and not balanced, and let $Q'$ be an acyclic approximation of $Q$. We shall prove that $T_{Q'}$ is homomorphically equivalent to $K_2^{\rightleftarrows}$, implying that

$Q'$ is equivalent to $Q_2^{\text{triv}}$. Observe that $T_{Q'}$ has no loops, otherwise $T_Q \to K_2^{\rightleftarrows} < T_{Q'}$, which contradicts the minimality of $T_{Q'}$. Since $T_{Q'}$ has no loops, it follows that it is bipartite and then $T_{Q'} \to K_2^{\rightleftarrows}$. Now we shall prove that $K_2^{\rightleftarrows} \to T_{Q'}$. Suppose that $K_2^{\rightleftarrows} \not\to T_{Q'}$, then $K_2^{\rightleftarrows}$ is not subgraph of $T_{Q'}$. This implies that $T_{Q'}$ is balanced (since $Q'$ is acyclic and $T_{Q'}$ does not have $K_2^{\rightleftarrows}$ as a subgraph) and since $T_Q \to T_{Q'}$, using Claim 4.3.2, we conclude that $T_Q$ is balanced too, which is a contradiction.

Finally, suppose that $T_Q$ is bipartite and balanced (or equivalently, $T_Q$ is just balanced) and let $Q'$ be an acyclic approximation of $Q$. Using the same argument that in the previous case, we know that $T_{Q'}$ has no loops and then $Q'$ is not equivalent to the trivial CQ. We shall prove that $K_2^{\rightleftarrows}$ is not a subgraph of $T_{Q'}$, concluding the result. Suppose that $K_2^{\rightleftarrows}$ is a subgraph of $T_{Q'}$. Since $T_Q$ is balanced, we have that $T_Q \to \vec{P}_k$, for some $k$ (from the Proof of Claim 4.3.2). Since $\vec{P}_k < K_2^{\rightleftarrows}$ ($\vec{P}_k$ is bipartite and $K_2^{\rightleftarrows}$ is not subgraph of $\vec{P}_k$), we have that $T_Q \to \vec{P}_k < T_{Q'}$, which contradicts the minimality of $T_{Q'}$. □

As a corollary to the proof of this result, we obtain:

**Corollary 4.3.3.** *Let $Q$ be a Boolean cyclic CQ over graphs. Then all minimized acyclic approximations of $Q$ have strictly fewer joins than $Q$.*

**Proof.** Let $Q'$ be a minimized acyclic approximation of $Q$. It suffices to show that the tableau $T_{Q'}$ has strictly fewer edges than the tableau $T_Q$. We denote by $|E(T_{Q'})|$ and $|T_{Q'}|$ the numbers of edges and vertices in $T_{Q'}$, respectively.

If $T_Q$ is not bipartite, or if is bipartite but not balanced, using Theorem 4.3.1 and the fact that $T_Q$ has at least 3 edges ($T_Q$ is cyclic), we have the result.

Now, if $T_Q$ is bipartite and balanced we know that $K_2^{\rightleftarrows} \not\subseteq T_{Q'}$ (Theorem 4.3.1) and since $T_{Q'}$ is core ($Q'$ is a minimized CQ), we know that $T_{Q'}$ is a homomorphic image of $T_Q$, via some $h$, that is, $T_Q \xrightarrow{h} T_{Q'}$, where $\text{Im}(h) = T_{Q'}$. It suffices to show that there are two edges in $T_Q$ such that are mapped via $h$ to the same edge in $T_{Q'}$. Since $T_Q$ is cyclic there exists a connected component of $T_Q$ which is cyclic, namely $H$ (connected in the sense that $H^u$ is connected). Let $h'$ be the restriction of $h$ to $H$. Note that $\text{Im}(h')$ is connected, acyclic and since $K_2^{\rightleftarrows} \not\subseteq \text{Im}(h')$, it follows that $|E(\text{Im}(h'))| = |\text{Im}(h')| - 1 \leq |H| - 1$. Finally, observe that $|E(H)| > |H| - 1$, otherwise since $H$ is connected, then $H$ would be acyclic, which is a contradiction. Thus, $|E(\text{Im}(h'))| < |E(H)|$ and $h'$ maps two edges to one edge in $T_{Q'}$. In particular, $h$ maps two edges to one edge in $T_{Q'}$. □

The most interesting case, according to the theorem above, is that of queries whose tableaux are bipartite and balanced. But then a natural question is whether under such restrictions, CQs are already tractable. It turns out that they are not, so it does make perfect sense to approximate queries from that class.

**Proposition 4.3.4.** *The combined complexity of evaluating Boolean CQs over graphs whose tableaux are bipartite and balanced is* NP*-complete.*

**Proof.** Note that any balanced digraph is bipartite, thus bipartite and balanced digraphs are exactly balanced digraphs. It suffices to prove (due to the correspondence between CQs and tableaux) that the following equivalent problem is NP-complete: Given a balanced digraph $G$ and a digraph $H$, check if $G \to H$.

Now we prove the complexity result. It is known, see [21, 18], that there exists an oriented tree $T$ (undirected tree plus orientation in the edges) such that the next problem is NP-complete: Given a digraph $G$, decide if $G \to T$. This problem remains NP-complete even for $G$ balanced, since a digraph homomorphic to $T$ must be balanced (using Claim 4.3.2 and the fact that any oriented tree is balanced). Thus the result follows and in fact the problem is NP-complete even for a fixed oriented tree. □

### 4.3.2 Non-Boolean queries

For CQs with free variables, it is still true that those whose tableaux are bipartite have nontrivial acyclic approximations. However, now some queries with non-bipartite tableaux may have approximations whose bodies do not trivialize to just $E(x, x)$. For example, consider a query $Q(x, y)$:–$E(x, y)$, $E(y, z)$, $E(z, x)$. It can be shown easily that $Q'(x, y)$:–$E(x, y)$, $E(y, x)$, $E(x, x)$ is an acyclic approximation of it; the tableau of $Q'$ is $K_2^{\rightleftarrows}$ with a loop on one of the nodes (recall that the definition of query acyclicity refers to tree decompositions of the query hypergraphs, so $Q'$ is indeed acyclic).

What distinguishes the case of bipartite tableaux now when we look at queries with free variables is that they do not have subgoals of the form $E(x, x)$ in approximations. That is, we have the following dichotomy:

**Theorem 4.3.5.** *Let $Q(\bar{x})$ be a cyclic CQ over graphs. If its tableau $T_Q$*

- *is not bipartite, then all of $Q$'s acyclic approximations have a subgoal of the form $E(x, x)$.*
- *is bipartite, then $Q$ has an acyclic approximation without a subgoal of the form $E(x, x)$.*

**Proof.** Suppose that the tableau $T_Q$ of $Q$ is not bipartite and let $Q'$ be an acyclic approximation of $Q$. Suppose that $T_{Q'}$ has no loops. Then $T_{Q'}$ is bipartite (since $T^u_{Q'}$ is acyclic), implying that $T_{Q'} \to K_2^{\rightleftarrows}$. Since $T_Q \to T_{Q'}$, we have that $T_Q \to K_2^{\rightleftarrows}$ as well. If follows that $T_Q$ is bipartite, which is impossible. Therefore, $T_{Q'}$ has loops, i.e, $Q'$ has a subgoal of the form $E(x, x)$.

Now, suppose that the tableau $T_Q$ of $Q$ is bipartite. Using an argument similar to the proof of Theorem 4.0.1 we shall prove that there exists an acyclic approximation of $Q$ without a subgoal of the form $E(x, x)$, i.e, whose tableau has no loops. Let $\mathcal{A}_Q$ be the set of all digraphs with distinguished elements $(H, \bar{u})$ such that $H^{\mathrm{u}}$ is acyclic and has no loops, $(T_Q, \bar{x}) \to (H, \bar{u})$ and $|H| \leq |T_Q|+1$ (where $|G|$ is the number of vertices of the digraph $G$). Clearly, $\mathcal{A}_Q$ is finite (up to isomorphism). Moreover, it is not empty, since there exists a homomorphism $h$ from $T_Q$ to $K_2^{\rightleftarrows}$ (this follows from the fact that $T_Q$ is bipartite) and this implies that $(K_2^{\rightleftarrows}, h(\bar{x}))$ is contained in $\mathcal{A}_Q$. We can pick a minimal element (with respect to $\to$) $(H', \bar{u}')$ from $\mathcal{A}_Q$. We shall show that $Q'(\bar{u}')$, the CQ whose tableau is $(H', \bar{u}')$, is an acyclic approximation of $Q$. Suppose not, then there exists an acyclic CQ $Q''(\bar{x}'')$ such that $(T_Q, \bar{x}) \xrightarrow{g} (T_{Q''}, \bar{x}'') < (H', \bar{u}')$. Observe that $(\mathrm{Im}(g), \bar{x}'')$ has no loops (otherwise, $(T_{Q''}, \bar{x}'') \not\to (H', \bar{u}')$), $\mathrm{Im}(g)^u$ is acyclic (since $\mathrm{Im}(g)$ is subgraph of $T_{Q''}$), $(T_Q, \bar{x}) \to (\mathrm{Im}(g), \bar{x}'')$ and $|\mathrm{Im}(g)| \leq |T_Q|$. Therefore, $(\mathrm{Im}(g), \bar{x}'')$ is contained in $\mathcal{A}_Q$. Finally, note that $(\mathrm{Im}(g), \bar{x}'') < (H', \bar{u}')$, which contradicts the minimality of $(H', \bar{u}')$ in $\mathcal{A}_Q$. Thus, $Q'$ is an acyclic approximation of $Q$ and since $T_{Q'} = H'$ has no loops, we have the result. □

We already know that acyclic approximations of an arbitrary query $Q$ can be constructed in single-exponential time and are bounded by the size of $Q$. For Boolean queries we saw that acyclic approximations also have strictly fewer joins than $Q$. With free variables, the number of joins may sometimes be the same as for $Q$ itself.

**Proposition 4.3.6.** *There is a non-Boolean cyclic CQ over graphs such that all of its minimized acyclic approximations have exactly as many joins as $Q$.*

**Proof.** Consider the following query:

$$Q(x_1, x_2, x_3) :- E(x_1, x_2), E(x_2, x_3), E(x_3, x_4), E(x_4, x_1).$$

This query is minimized. Its tableau, which we denote by $(G, x, y, z)$, contains an oriented cycle on nodes $x, y, z, u$, with $x, y, z$ being distinguished nodes.

Let $G' = \langle V', E' \rangle$ be a graph containing nodes $x'_1, x'_2, x'_3$ (not necessarily distinct) so that $(G', x'_1, x'_2, x'_3)$ is a tableau of an acyclic approximation of $Q$. We know that $(G', x'_1, x'_2, x'_3)$ is an image of some homomorphism $h$ defined on $G$. Note that if the homomorphism $h$ were one-to-one, then all the edges of $G$ would be present in $G'$ and thus $G'$ would be cyclic. Hence, $|V'| \leq 3$.

By definition, $h(x_i) = x'_i$ for $1 \leq i \leq 3$. Consider first a homomorphism $h$ so that $x'_i$s, for $1 \leq i \leq 3$, are distinct. Then there are two possibilities where $x_4$ could be mapped:

- If $h(x_4) = x'_1$ or $h(x_4) = x'_3$, then $Im(h)$ is a cyclic graph, contradicting the assumption.

- If $h(x_4) = x'_2$, we get a graph consisting of two copies of $K_2^{\rightleftarrows}$, i.e., a graph $G_0$ with nodes $x'_i$, $1 \leq i \leq 3$, and edges $(x'_1, x'_2), (x'_2, x'_1)$ as well as $(x'_2, x'_3), (x'_3, x'_2)$. The corresponding query $Q_0(x'_1, x'_2, x'_3)$:–$E(x'_1, x'_2), E(x'_2, x'_1), E(x'_2, x'_3), E(x'_3, x'_2)$ has the same 3 joins as the original query.

Next we see what happens when $h$ collapses some of $x_i$'s, for $1 \leq i \leq 3$. First we look at the cases when $h$ collapses two of those, so that its image has two nodes. Suppose we collapse $x_1$ and $x_2$, i.e., $h(x_1) = h(x_2)$ (and thus $x'_1 = x'_2$) and $x'_3 \neq x'_1$. There are three possibilities for $x_4$:

- If $h(x_4) = x'_1 = x'_2$, then $Im(h)$ is the graph $G_1$ with nodes $x'_1, x'_3$ and edges $(x'_1, x'_3)$, $(x'_3, x'_1)$ , $(x'_1, x'_1)$, with distinguished nodes $(x'_1, x'_1, x'_3)$. It is routine to check that $(G_0, x'_1, x'_2, x'_3) < (G_1, x'_1, x'_1, x'_3)$, and hence the result of this homomorphism is not an acyclic approximation.

- If $h(x_4) = x'_3$, then the image of $h$ is a graph with 4 edges, hence corresponding to a query with 3 joins, same as in the original $Q$.

- If $h(x_4)$ is different from $x'_1, x'_2, x'_3$, then $Im(h)$ has a cycle.

The case when $x_2$ and $x_3$ are collapsed to the same node by $h$ is completely symmetric. Now assume that $h$ collapses $x_1$ and $x_3$, i.e., $x'_1 = x'_3$. Again there are three cases.

- If $h(x_4) = x'_2$, then the image of $h$ is $(G_2, x'_1, x'_2, x'_1)$ where $G_2$ is a copy of $K_2^{\rightleftarrows}$ on $x'_1, x'_2$. In this case again we easily verify $(G_0, x'_1, x'_2, x'_3) < (G_2, x'_1, x'_2, x'_1)$, meaning that the latter cannot be an acyclic approximation.

- If $h(x_4) = x'_1$, then the image of the original tableau is the same graph as in the previous case, plus a loop on $x'_1$. Hence, the same argument as above shows that it cannot be an acyclic approximation.

- Otherwise, if $h(x_4)$ is different from $x'_1, x'_2, x'_3$, then $Im(h)$ is a union of two copies of $K_2^{\rightleftarrows}$, and thus it has the same 3 joins as the original query.

Finally, if $h$ collapses all nodes (say to $x'_1$), there are two possibilities.

- If $h(x_4) = x'_1$, then we end up with a trivial query with the tableau $(K_1^{\circlearrowleft}, x'_1, x'_1, x'_1)$, and clearly $(G_0, x'_1, x'_2, x'_3) < (K_1^{\circlearrowleft}, x'_1, x'_1, x'_1)$.

- If $h(x_4) = x'_4 \neq x'_1$, then $Im(h)$ consists of a $K_2^{\rightleftarrows}$ on $x'_1$ and $x'_4$, as well as a loop on $x'_1$. In this case we again easily verify that $(G_0, x'_1, x'_2, x'_3) < Im(h)$.

Hence, in all the cases when the number of joins is reduced, the resulting query is not an acyclic approximation, which proves the proposition. □

## 4.4 Bounded treewidth approximations

We have already seen that treewidth-$k$ approximations of a CQ $Q$ always exist, that they cannot exceed the size of $Q$, and can be constructed in single-exponential time. There is an analog of the dichotomy for acyclic queries, in which bipartiteness (i.e., being 2-colorable) is replaced by $(k+1)$-colorability for $\mathsf{TW}(k)$.

**Theorem 4.4.1.** *Let $Q$ be a CQ over graphs. If its tableau $T_Q$*

- *is not $(k+1)$-colorable, then all of its $\mathsf{TW}(k)$-approximations have a subgoal of the form $E(x,x)$;*
- *is $(k+1)$-colorable, then $Q$ has a $\mathsf{TW}(k)$-approximation without a subgoal of the form $E(x,x)$.*

**Proof.** We use similar arguments to the proof of Theorem 4.3.5. Recall that every digraph without loops of treewidth at most $k$ is $(k+1)$-colorable.

Suppose that the tableau $T_Q$ of $Q$ is not $(k+1)$-colorable and let $Q'$ be a $\mathsf{TW}(k)$-approximation of $Q$. Suppose that $T_{Q'}$ has no loops. Then $T_{Q'}$ is $(k+1)$-colorable, implying that $T_{Q'} \to K^{\rightleftarrows}_{k+1}$. Since $T_Q \to T_{Q'}$, we have that $T_Q \to K^{\rightleftarrows}_{k+1}$ as well. If follows that $T_Q$ is $(k+1)$-colorable, which is impossible. Therefore, $T_{Q'}$ have loops, i.e, $Q'$ has a subgoal of the form $E(x,x)$.

Now, suppose that the tableau $T_Q$ of $Q$ is $(k+1)$-colorable and let $\mathcal{A}_Q$ be the set of all digraphs with distinguished elements $(H,\bar{u})$ such that $H^{\mathrm{u}}$ has treewidth at most $k$ and has no loops, $(T_Q,\bar{x}) \to (H,\bar{u})$ and $|H| \leq |T_Q| + k$. Clearly, $\mathcal{A}_Q$ is finite (up to isomorphism). Moreover, it is not empty, since there exists a homomorphism $h$ from $T_Q$ to $K^{\rightleftarrows}_{k+1}$ ($T_Q$ is $(k+1)$-colorable) and this implies that $(K^{\rightleftarrows}_{k+1}, h(\bar{x}))$ is contained in $\mathcal{A}_Q$. We can pick a minimal element (with respect to $\to$) $(H',\bar{u}')$ from $\mathcal{A}_Q$. We shall show that $Q'(\bar{u}')$, the CQ whose tableau is $(H',\bar{u}')$, is a $\mathsf{TW}(k)$-approximation of $Q$. Suppose not, then there exists a CQ $Q''(\bar{x}'')$ with treewidth at most $k$ such that $(T_Q,\bar{x}) \xrightarrow{g} (T_{Q''},\bar{x}'') < (H',\bar{u}')$. Observe that $(\mathrm{Im}(g),\bar{x}'')$ has no loops (otherwise, $(T_{Q''},\bar{x}'') \not\to (H',\bar{u}')$), $\mathrm{Im}(g)^u$ has treewidth at most $k$ (since $\mathrm{Im}(g)$ is subgraph of $T_{Q''}$), $(T_Q,\bar{x}) \to (\mathrm{Im}(g),\bar{x}'')$ and $|\mathrm{Im}(g)| \leq |T_Q|$. Therefore, $(\mathrm{Im}(g),\bar{x}'')$ is contained in $\mathcal{A}_Q$. Finally, note that $(\mathrm{Im}(g),\bar{x}'') < (H',\bar{u}')$, which is a contradiction with the minimality of $(H',\bar{u}')$ in $\mathcal{A}_Q$. Thus, $Q'$ is a $\mathsf{TW}(k)$-approximation of $Q$ and since $T_{Q'} = H'$ has no loops, we have the result. □

Recall that a Boolean CQ $Q^{\mathrm{triv}}() \text{:-} E(x,x)$ is a trivial (acyclic, or treewidth-$k$) approximation of every Boolean CQ. In the acyclic case, 2-colorability (or bipartiteness) of $T_Q$ was equivalent to the existence of nontrivial approximations. This result extends to treewidth-$k$.

**Corollary 4.4.2.** *A Boolean CQ $Q$ over graphs has a nontrivial* $\mathsf{TW}(k)$*-approximation iff its tableau $T_Q$ is $(k+1)$-colorable.*

**Proof.** Suppose that the tableau $T_Q$ of $Q$ is not $(k+1)$-colorable and let $Q'$ be a $\mathsf{TW}(k)$-approximation of $Q$. Using Theorem 4.4.1 we know that $T_{Q'}$ has loops. Therefore, $Q'$ is equivalent to $Q^{\text{triv}}$.

Now, suppose that the tableau $T_Q$ of $Q$ is $(k+1)$-colorable. Assume that $T_{Q'}$ has loops. Then we have that $T_Q \to K^{\rightleftarrows}_{k+1} < T_{Q'}$, which is a contradiction with the minimality of $T_{Q'}$. Then $T_{Q'}$ has no loops, i.e., is not equivalent to $Q^{\text{triv}}$. □

Note the big difference in the complexity of testing for the existence of nontrivial approximations: while it is in PTIME in the acyclic case, the problem is already NP-complete for $\mathsf{TW}(2)$.

# Chapter 5

# Complexity of approximations

In this chapter we do a complete complexity analysis of the problem of identifying approximations. We begin with the case of acyclic approximations, capturing exactly the complexity of this problem. Next, we study the case of bounded treewidth approximations. It turns out that there is a fundamental difference of complexity between these two cases.

## 5.1 Acyclic approximations

We have seen that a (minimized) acyclic approximation can be found in single-exponential time. Of course this is expected given NP-hardness of most static analysis tasks related to CQs. To do a more detailed analysis of complexity, we formulate it as the following decision problem:

| | |
|---|---|
| Problem: | Acyclic Approximation |
| Input: | a cyclic CQ $Q$, an acyclic CQ $Q'$. |
| Question: | Is $Q'$ an acyclic approximation of $Q$? |

To solve Acyclic Approximation, we need to check two things:

1. $Q' \subseteq Q$; and
2. there is no acyclic $Q''$ such that $Q' \subset Q'' \subseteq Q$.

The first subproblem is solvable in NP. Checking whether there is an acyclic query $Q''$ not equivalent to $Q'$ with $Q' \subseteq Q'' \subseteq Q$ is solvable in NP too. This means $T_Q \to T_{Q''} < T_{Q'}$ and hence such $Q''$, if it exists can always be chosen not to exceed the size of $Q$. Therefore, one can guess $T_{Q''}$ and all homomorphisms in NP. Furthermore, since both $T_{Q''}$ and $T_{Q'}$

are acyclic, checking that $T_{Q'} \not\to T_{Q''}$ can be done in polynomial time. Thus, the second subproblem is solvable in coNP.

Hence, to solve Acyclic Approximation, we need to solve an NP subproblem and a coNP subproblem. This means that the problem is in complexity class DP: this is the class of problems (languages) which are intersections of an NP language and a coNP language [30]. It turns out that the problem is also hard for the class.

**Theorem 5.1.1.** *The problem* Acyclic Approximation *is* DP*-complete. It remains* DP*-complete even if $Q'$ is fixed and both $Q$ and $Q'$ are Boolean and minimized (i.e., their tableaux are cores).*

DP-completeness appeared in database literature in connections with cores: checking whether $G' = \text{core}(G)$, for two graphs $G$ and $G'$, is known to be DP-complete [9]. The source of DP-completeness in our case is completely different, as hardness applies even if the tableaux of queries are cores to start with, and the proof, which is quite involved, uses techniques different from those in [9].

We shall prove that the following equivalent problem is DP-complete: Given a cyclic digraph $G$ and an acyclic digraph $T$, check if $G \to T$ and there is no acyclic digraph $T'$ such that $G \to T' < T$. We define a weaker version of this problem: Given a cyclic digraph $G$ and an acyclic digraph $T$, check if $G \to T$ and there is no proper subgraph $T'$ of $T$ such that $G \to T' < T$.

First, we shall prove the following:

**Proposition 5.1.2.** *The weaker version of the Acyclic Approximation Problem is DP-complete.*

We need some definitions and claims. Consider the oriented paths $P_i = 0^{i+1}10^{11-i}$, for each $1 \leq i \leq 9$. Observe that all these oriented paths are incomparable cores and have net length 11. We have the following:

**Claim 5.1.3.** *For each $1 \leq i < j \leq 9$, there exists an oriented path $P_{ij}$ such that $P_{ij} \to P_i$, $P_{ij} \to P_j$ and $P_{ij} \not\to P_k$ for each $1 \leq k \leq 9$, $k \neq i, j$.*

**Proof.** We just take $P_{ij} = 0^{i+1}100^{j-i}10^{11-j}$. Using Lemma 4.2.4 it can be verified that the conditions are satisfied. □

**Claim 5.1.4.** *For each $1 \leq i < j < k \leq 9$, there exists an oriented path $P_{ijk}$ such that $P_{ijk} \to P_i$, $P_{ijk} \to P_j$, $P_{ijk} \to P_k$ and $P_{ijk} \not\to P_l$ for each $1 \leq l \leq 9$, $l \neq i, j, k$.*

**Proof.** We just take $P_{ijk} = 0^{i+1}100^{j-i}100^{k-j}10^{11-k}$. Using Lemma 4.2.4 it can be verified that the conditions are satisfied. □

We define a digraph $Q^*$ as follows: Consider the balanced cycle $[a_1, a_2, ..., a_8, a_1]$ defined by the string 01010101. For each $1 \leq i \leq 8$, add a disjoint copy of $P_i$. If $i$ is odd, identify $a_i$ with the terminal vertex of $P_i$; otherwise, identify it with the initial vertex of $P_i$. Finally add two new vertices $x$ and $y$, and two new edges: from $x$ to the initial vertex of the copy of $P_1$ and from the terminal vertex of the copy of $P_8$ to $y$. Note that $hg(Q^*) = 25$ and the vertices $x$ and $y$ are the only ones in $Q^*$ with level 0 and 25, respectively. See Figure 5.1. In Figure 5.2 we can see some of the levels of the vertices.

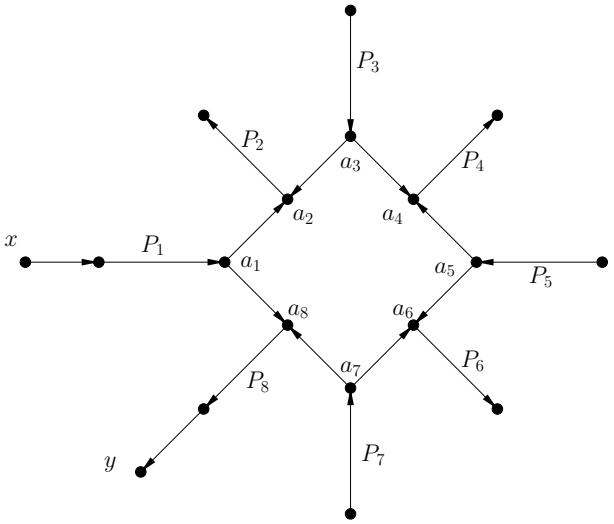


Figure 5.1: The digraph $Q^*$.

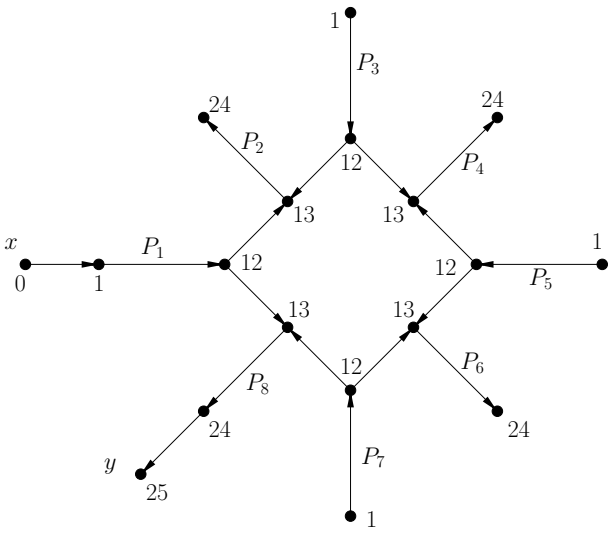


Figure 5.2: The digraph $Q^*$ and some of its levels.

Now, we define digraphs $T_1, T_2, T_3$ and $T_4$ as follows: Each $T_i$ for $1 \leq i \leq 4$ is obtained from $Q^*$ by identifying some specific vertices. For $T_1$ identify $a_1$, $a_2$ and $a_3$ with $a_7$, $a_6$ and $a_5$, respectively. In the case of $T_2$ identify $a_8$, $a_1$ and $a_2$ with $a_6$, $a_5$ and $a_4$, respectively. For $T_3$ identify $a_7$, $a_8$ and $a_1$ with $a_5$, $a_4$ and $a_3$, respectively. Finally, for $T_4$ identify $a_6$, $a_7$ and $a_8$ with $a_4$, $a_3$ and $a_2$. Note that for each $1 \leq i \leq 4$, $hg(T_i) = 25$ and the vertices $x_i$ and $y_i$ are the only ones in $T_i$ with level 0 and 25, respectively. See Figure 5.3 and 5.4.

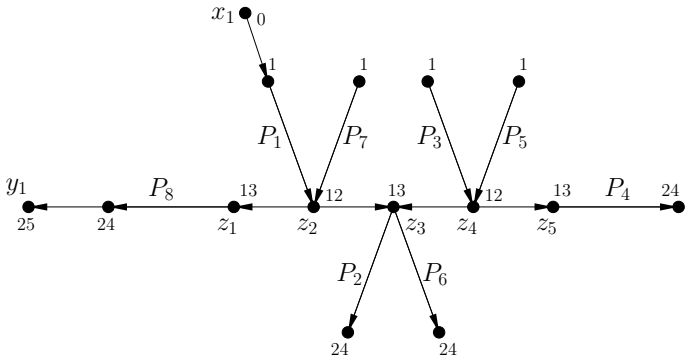


Figure 5.3: The digraph $T_1$ and some of its levels.

Using Lemma 4.2.4 and the incomparability of the $P_i$'s, it easily follows that $T_1, T_2, T_3$ and $T_4$ are incomparable cores. Furthermore, observe that $Q^* \xrightarrow{h_i} T_i$ for all $1 \leq i \leq 4$ (Using Claim 4.2.7). Note that for each $1 \leq i \leq 4$, $h_i$ is a surjective homomorphism, i.e., $\mathrm{Im}(h_i) = T_i$. In fact we have the following:

**Claim 5.1.5.** *For each $1 \leq i \leq 4$, $h_i$ is the unique homomorphism from $Q^*$ into $T_i$. In particular, any homomorphism from $Q^*$ to $T_i$ is surjective, for each $1 \leq i \leq 4$.*

**Proof.** Let $h$ be a homomorphism that witnesses $Q^* \to T_1$. We shall prove that $h = h_1$. Since $h$ preserves levels (Lemma 4.2.4), we know that either $h(a_8) = z_1$, $h(a_8) = z_3$ or $h(a_8) = z_5$. If $h(a_8) = z_3$, then $P_8 \to P_2$ or $P_8 \to P_6$, so this is impossible. Similarly, if $h(a_8) = z_5$, then $P_8 \to P_4$, which is impossible too. It follows that $h(a_8) = z_1 = h_1(a_8)$. Using the same argument ($h$ preserves levels and the incomparability of $P_i$'s) it easily follows that $h = h_1$. For $i = 2, 3, 4$ the argument is analogous. □

A key property of these digraphs $T_i$, $1 \leq i \leq 4$, is the following:

**Claim 5.1.6.** *For each $1 \leq i \leq 4$, $Q^* \to T_i$ and there is no acyclic $T'$ such that $Q^* \to T' < T_i$.*

**Proof.** By contradiction, suppose that there exists an acyclic digraph $T'$ such that $Q^* \xrightarrow{h} T' \xrightarrow{g} T_1$ and $T_1 \not\to T'$. If $h(a_1) = h(a_7), h(a_2) = h(a_6)$ and $h(a_3) = h(a_5)$, using Claim

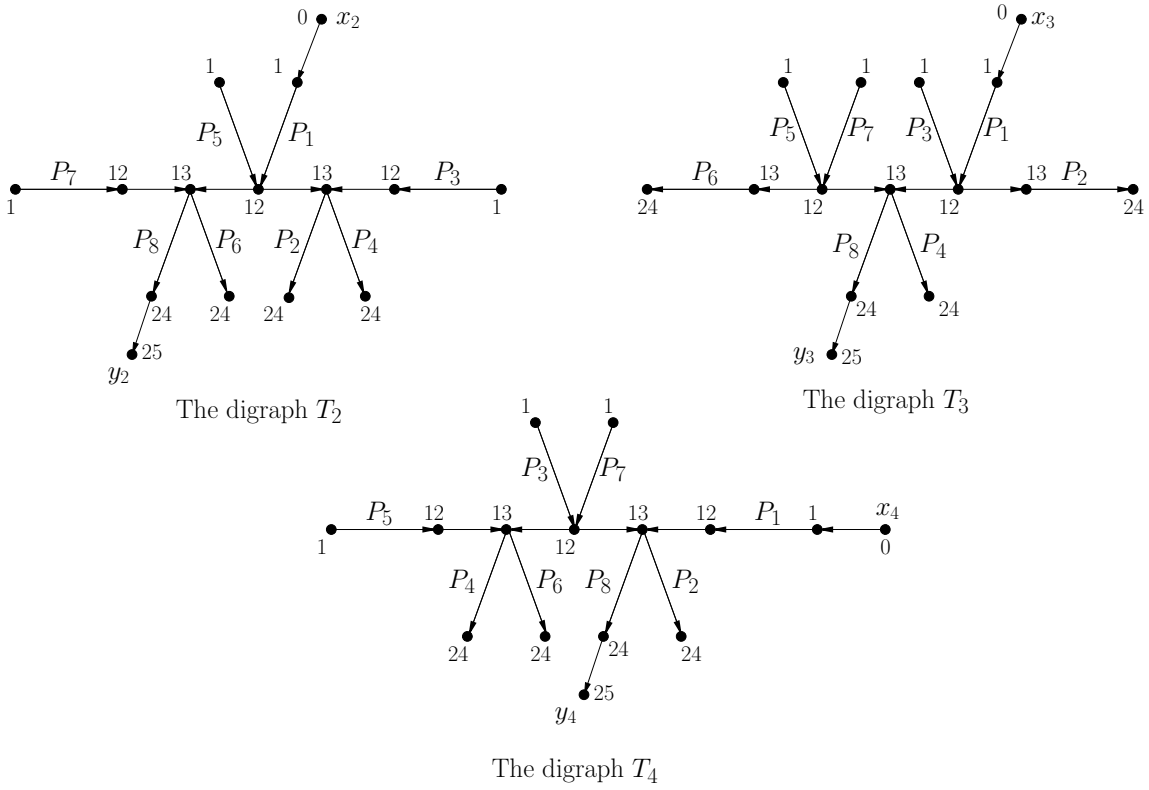


Figure 5.4: The digraphs $T_2$, $T_3$ and $T_4$.

4.2.8, it follows that $T_1 \to T'$, which is a contradiction. Thus, we have that either $h(a_1) \neq h(a_7), h(a_2) \neq h(a_6)$ or $h(a_3) \neq h(a_5)$. Using Claim 5.1.5, we have that $g \circ h = h_1$. Since the sets $\{h_1(a_8)\}$, $\{h_1(a_1), h_1(a_7)\}$, $\{h_1(a_2), h_1(a_6)\}$, $\{h_1(a_3), h_1(a_5)\}$ and $\{h_1(a_4)\}$ are disjoint, the sets $\{h(a_8)\}$, $\{h(a_1), h(a_7)\}$, $\{h(a_2), h(a_6)\}$, $\{h(a_3), h(a_5)\}$ and $\{h(a_4)\}$ are disjoint as well. Since $h(a_1) \neq h(a_7), h(a_2) \neq h(a_6)$ or $h(a_3) \neq h(a_5)$, necessarily $T'$ has an oriented cycle, which is a contradiction. For $i = 2, 3, 4$ the argument is analogous. □

Now, define $T_5$ as follows: Consider the disjoint union of $P_1$ and $P_8$. Add two new vertices $x_5$ and $y_5$ and three new edges: from $x_5$ to the initial vertex of $P_1$, from the terminal vertex of $P_1$ to the initial vertex of $P_8$ and from the terminal vertex of $P_8$ to $y_5$. Finally, add two disjoint copies of $P_9$ and identify the terminal vertex of one copy with the terminal vertex of $P_1$ and the initial vertex of the other copy with the initial vertex of $P_8$. See Figure 5.5.

Using Lemma 4.2.4 and the incomparability of the $P_i$'s, easily follows that $T_5$ is incomparable with $T_1, T_2, T_3, T_4$ and $Q^*$.

**Claim 5.1.7.** *For $(i, j) \in \{(1, 5), (2, 5), (3, 5), (1, 2), (1, 3), (2, 3)\}$, there exists an acyclic digraph $T_{ij}$ such that $T_{ij} \to T_i$, $T_{ij} \to T_j$ and $T_{ij} \not\to T_k$ for $1 \leq k \leq 5$, $k \neq i, j$.*

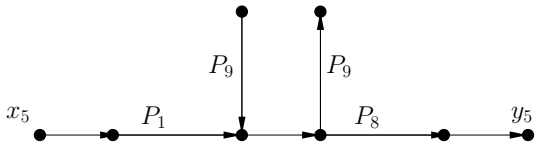


Figure 5.5: The digraph $T_5$.

**Proof.** Consider the oriented path $P$ constructed as follows: Take the disjoint union of $P_1$ and $P_8$. Add two new vertices $p_1$ and $p_2$, and three new edges: from $p_1$ to the initial vertex of $P_1$, from the terminal vertex of $P_1$ to the initial vertex of $P_8$ and from the terminal vertex of $P_8$ to $p_2$. Each $T_{ij}$ is obtained from $P$ by adding a disjoint copy of a specific path $X_{ij}$ and identifying the terminal vertex of $X_{ij}$ with the terminal vertex of $P_1$, see Figure 5.6. The $X_{ij}$'s are: $X_{15} = P_{79}, X_{25} = P_{59}, X_{35} = P_{39}, X_{12} = P_{57}, X_{13} = P_{37}$ and $X_{23} = P_{35}$. From Lemma 4.2.4 and Claim 5.1.3 it follows that this $T_{ij}$ satisfies the required conditions. □

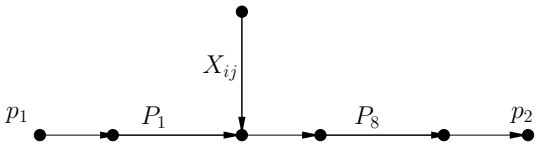


Figure 5.6: The structure of the $T_{ij}$'s.

**Claim 5.1.8.** *For $(i,j,k) \in \{(1,2,5),(2,4,5),(3,4,5)\}$, there exists an acyclic digraph $T_{ijk}$ such that $T_{ijk} \to T_i$, $T_{ijk} \to T_j$, $T_{ijk} \to T_k$ and $T_{ijk} \not\to T_l$ for $1 \leq l \leq 5$, $l \neq i,j,k$.*

**Proof.** Consider the oriented path $P$ as in the proof of Claim 5.1.7. The digraph $T_{125}$ is obtained from $P$ by adding a disjoint copy of $P_{579}$ and identifying the terminal vertex of $P_{579}$ with the terminal vertex of $P_1$. The others $T_{ijk}$'s are obtained from $P$ by adding a disjoint copy of a path $X_{ijk}$ and identifying the initial vertex of $X_{ijk}$ with the initial vertex of $P_8$. The $X_{ijk}$'s are: $X_{245} = P_{269}$ and $X_{345} = P_{249}$, see Figure 5.7 . From Lemma 4.2.4 and Claim 5.1.4, it follows that this $T_{ijk}$ satisfies the required conditions. □

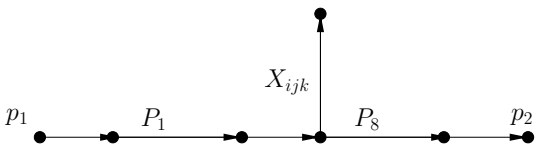


Figure 5.7: The structure of the $T_{245}$ and $T_{345}$.

We introduce some notation. Consider a digraph $G$ with two distinguished vertices $i_1$ and $t_1$. We call them the initial and terminal vertex of $G$, respectively, and we assume that they are distinct. Similarly, consider a digraph $H$ with $i_2$ and $t_2$ as its initial and terminal vertex, respectively. We define the concatenation $G \cdot H$ to be the digraph obtained from the disjoint union of $G$ and $H$ identifying $t_1$ with $i_2$. The initial and terminal vertex of $G \cdot H$ is $i_1$ and $t_2$, respectively. Finally, we define $G^{-1}$ to be the digraph $G$ with initial vertex $t_1$ and terminal vertex $i_1$.

We define the initial vertex of $Q^*$ as $x$ and its terminal vertex as $y$. For each $1 \leq i \leq 5$, we define $x_i$ and $y_i$ to be the initial and terminal vertex of $T_i$, respectively. Similarly, for each $T_{ij}$'s and $T_{ijk}$'s of Claims 5.1.7 and 5.1.8, we define the initial and terminal vertex to be the only vertex with level 0 and 25, respectively. In all Figures, an edge $uv$ labeled with a digraph $G$ with initial vertex $i$ and terminal vertex $t$, is understood as deletion of arc $uv$, union of a disjoint copy of $G$ and identification of $i$ with $u$ and $b$ with $t$.

Let $T$ be the digraph constructed as follows: Consider the disjoint union of $T_1 \cdot T_5^{-1}$, $T_2 \cdot T_5^{-1}$, $T_3 \cdot T_5^{-1}$ and $T_4 \cdot T_5^{-1}$, and identify their initial vertices with a new vertex $v$. Observe that $hg(T) = 25$ and the only vertices with level 0 are $v, u_1, u_2, u_3$ and $u_4$, and the only vertices with level 25 are $t_1, t_2, t_3$ and $t_4$. See Figure 5.8.

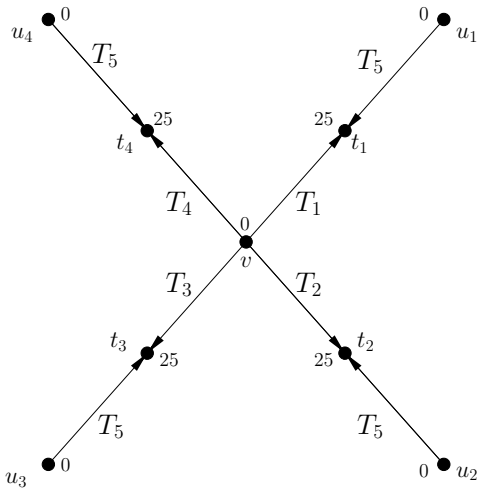


Figure 5.8: The digraph $T$ and some of its levels.

Now we recall the $(i, j)$-*chooser* notion defined in [21].

**Definition 5.1.9.** *Let $X = \{1, 2, 3\}$ and $i, j \in X$ be indices. An $(i, j)$-chooser is a digraph $T^*$ with two distinguished vertices $a$ and $b$ such that:*

- *For every homomorphism $h : T^* \to T$, we have $h(a) = t_1$ and $h(b) \neq t_i$, or $h(a) = t_2$ and $h(b) \neq t_j$.*

- *For every $k \in X$, $k \neq i$, there is a homomorphism $h : T^* \to T$ such that $h(a) = t_1$ and $h(b) = t_k$.*
- *For every $k \in X$, $k \neq j$, there is a homomorphism $h : T^* \to T$ such that $h(a) = t_2$ and $h(b) = t_k$.*

We define an *extended $(i, j)$-chooser* as in Definition 5.1.9 but considering $X = \{1, 2, 3, 4\}$. We have the following:

**Claim 5.1.10.** *There exists a $(1, 3)$-chooser $S_{13}$, a $(2, 1)$-chooser $S_{21}$, and a $(3, 2)$-chooser $S_{32}$.*

**Proof.** Exactly the same as Lemma 4 in [21], identifying of course $P_1, P_2, P_3$ and $P_4$, with $T_1, T_2, T_3$ and $T_5$, and the corresponding $P_{ij}$'s with the $T_{ij}$'s of Claim 5.1.7. □

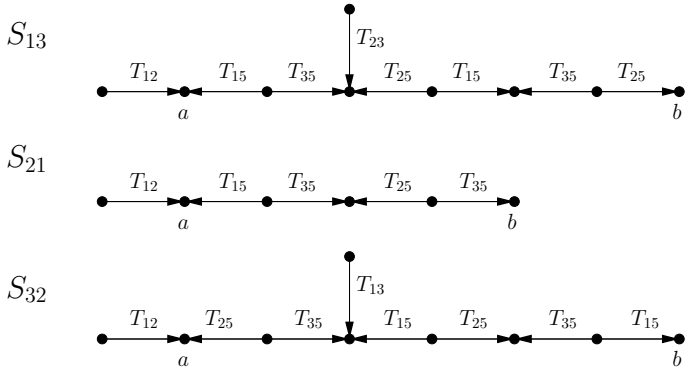


Figure 5.9: The choosers $S_{13}$, $S_{21}$ and $S_{31}$.

We rename the vertices $a$ and $b$ in $S_{13}, S_{21}$ and $S_{32}$ to $a_1, b_1$; $a_2, b_2$ and $a_3, b_3$, respectively.

**Claim 5.1.11.** *There exists an extended $(2, 1)$-chooser $\tilde{S}_{21}$ and an extended $(3, 4)$-chooser $\tilde{S}_{34}$.*

**Proof.** Define $\tilde{S}_{21} = T_{12} \cdot T_{125}^{-1} \cdot T_{345}$ and $\tilde{S}_{34} = T_{12} \cdot T_{25}^{-1} \cdot T_{35} \cdot T_{15}^{-1} \cdot T_{245} \cdot T_{35}^{-1} \cdot T_{15}$. For $\tilde{S}_{21}$ and $\tilde{S}_{34}$ let $a$ to be the terminal vertex of theirs copies of $T_{12}$ and $b$ to be theirs terminal vertices. See Figure 5.10 and 5.11.

Consider a homomorphism $h : \tilde{S}_{21} \to T$. Necessarily, we have either $h(a) = t_1$ or $h(a) = t_2$. Suppose $h(a) = t_1$. Then, either $h(x_1) = u_1$ or $h(x_1) = v$. If $h(x_1) = u_1$, then $h(b) = t_1$. If $h(x_1) = v$, then either $h(b) = t_3$ or $h(b) = t_4$. Thus, if $h(a) = t_1$, then $h(b) \neq t_2$, and all the combinations $h(a) = t_1, h(b) = t_1$; $h(a) = t_1, h(b) = t_3$ and $h(a) = t_1, h(b) = t_4$ are possible. Suppose now that $h(a) = t_2$. Then, either $h(x_1) = u_2$ or $h(x_1) = v$. If $h(x_1) = u_2$,

then $h(b) = t_2$. If $h(x_1) = v$, then either $h(b) = t_3$ or $h(b) = t_4$. Thus, if $h(a) = t_2$, then $h(b) \neq t_1$, and all the combinations $h(a) = t_2, h(b) = t_2$; $h(a) = t_2, h(b) = t_3$ and $h(a) = t_2, h(b) = t_4$ are possible. Therefore, $\tilde{S}_{21}$ is an extended $(2,1)$-chooser.

Consider a homomorphism $h : \tilde{S}_{34} \to T$. Suppose $h(a) = t_1$. Then $h(x_1) = u_1$ and $h(x_2) = t_1$. Now, we have either $h(x_3) = u_1$ or $h(x_3) = v$. If $h(x_3) = u_1$, then $h(x_4) = t_1$, $h(x_5) = u_1$ and $h(b) = t_1$. If $h(x_3) = v$, then $h(x_4) = t_2$ or $h(x_4) = t_4$. If $h(x_4) = t_2$, then $h(x_5) = u_2$ and $h(b) = t_2$. If $h(x_4) = t_4$, then $h(x_5) = u_4$ and $h(b) = t_4$. Thus, if $h(a) = t_1$, then $h(b) \neq t_3$, and all the combinations $h(a) = t_1, h(b) = t_1$; $h(a) = t_1, h(b) = t_2$ and $h(a) = t_1, h(b) = t_4$ are possible. Now, suppose that $h(a) = t_2$. Then, we have either $h(x_1) = u_2$ or $h(x_1) = v$. If $h(x_1) = u_2$, then $h(x_2) = t_2$, $h(x_3) = u_2$, $h(x_4) = t_2$, $h(x_5) = u_2$ and $h(b) = t_2$. If $h(x_1) = v$, then $h(x_2) = t_3$, $h(x_3) = u_3$ and $h(x_4) = t_3$. Now, we have either $h(x_5) = u_3$ or $h(x_5) = v$. If $h(x_5) = u_3$, then $h(b) = t_3$. If $h(x_5) = v$, then $h(b) = t_1$. Thus, if $h(a) = t_2$, then $h(b) \neq t_4$, and all the combinations $h(a) = t_2, h(b) = t_1$; $h(a) = t_2, h(b) = t_2$ and $h(a) = t_2, h(b) = t_3$ are possible. Therefore, $\tilde{S}_{34}$ is an extended $(3,4)$-chooser. □

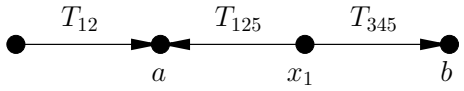


Figure 5.10: The extended $(2,1)$-chooser $\tilde{S}_{21}$.

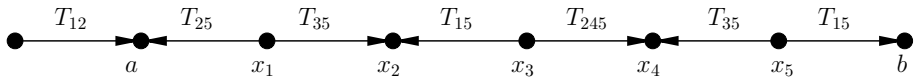


Figure 5.11: The extended $(3,4)$-chooser $\tilde{S}_{34}$.

Let $G$ be a digraph and $u \in V(G)$ ($V(G)$ and $E(G)$ denote the sets of vertices and edges of $G$, respectively). We define the digraph $G - u$ to be $(V(G) \setminus \{u\}, E(G) \setminus \{(s,t) : t = u \text{ or } s = u\})$. Observe from the proof of Claim 5.1.11 that the image of any homomorphism from $\tilde{S}_{21}$ or $\tilde{S}_{34}$ into $T$ has vertices in common with the subgraph $T_4 \cdot T_5^{-1} - v$ of $T$, only when $h(b) = t_4$. Thus, we have the following:

**Claim 5.1.12.** *Let $h$ be a homomorphism from either $\tilde{S}_{21}$ or $\tilde{S}_{34}$ into $T$. If $h(b) \neq t_4$, then the homomorphic image of $h$ does not have vertices in common with the subgraph $T_4 \cdot T_5^{-1} - v$ of $T$.*

We define $T'$ as in [21]: Take the disjoint union of $S_{13}$, $S_{21}$ and $S_{32}$ and identify $b_1$, $b_2$ and $b_3$ into a new vertex $b$. For the choosers and the extended choosers we let $a$ be the initial vertex and $b$ be the terminal vertex. See Figure 5.12.

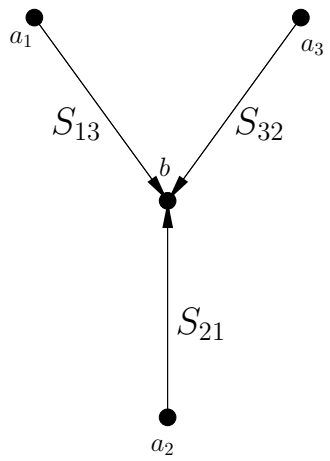


Figure 5.12: The digraph $T'$.

We have the following:

**Claim 5.1.13.** *There is no homomorphism $h : T' \to T$ such that $h(a_1) = h(a_2) = h(a_3)$. Furthermore, for any triple $(t_i, t_j, t_k) \in \{t_1, t_2\}^3 \setminus \{(t_1, t_1, t_1), (t_2, t_2, t_2)\}$ there exists a homomorphism $h : T' \to T$ such that $h(a_1) = t_i$, $h(a_2) = t_j$ and $h(a_3) = t_k$.*

**Proof.** Exactly the same as Lemma 5 in [21]. □

Now we construct our main gadget $\tilde{T}$ as follows: Consider two vertices $p$ and $q$, and add two disjoint copies of $T'$, namely, $T'_1$ and $T'_2$. We rename the vertices $a_1, a_2, a_3$ and $b$ in $T'_1$ to $a_1^1, a_2^1, a_3^1$ and $b^1$, and the vertices $a_1, a_2, a_3$ and $b$ in $T'_2$ to $a_1^2, a_2^2, a_3^2$ and $b^2$. Add three disjoint copies of $\tilde{S}_{21}$, namely, $\tilde{S}_{21}^1, \tilde{S}_{21}^2$ and $\tilde{S}_{21}^3$, and for each $1 \leq i \leq 3$ rename the vertices $a$ and $b$ in $\tilde{S}_{21}^i$ to $a_{21}^i$ and $b_{21}^i$. Now, identify $b_{21}^1$ and $b_{21}^2$ with $p$, and $b_{21}^3$ with $q$. Also, for each $1 \leq i \leq 3$ identify $a_{21}^i$ with $a_i^1$.

Similarly, add three disjoint copies of $\tilde{S}_{34}$, namely, $\tilde{S}_{34}^1, \tilde{S}_{34}^2$ and $\tilde{S}_{34}^3$, and for each $1 \leq i \leq 3$ rename the vertices $a$ and $b$ in $\tilde{S}_{34}^i$ to $a_{34}^i$ and $b_{34}^i$. Now, identify $b_{34}^1$ and $b_{34}^2$ with $p$, and $b_{34}^3$ with $q$. Also, for each $1 \leq i \leq 3$ identify $a_{34}^i$ with $a_i^2$. See Figure 5.13.

We have the following:

**Claim 5.1.14.** *There is no homomorphism $h : \tilde{T} \to T$ such that $h(p) = h(q)$. Furthermore, for any pair $(t_i, t_j) \in \{t_1, t_2, t_3, t_4\}^2 \setminus \{(t_1, t_1), (t_2, t_2), (t_3, t_3), (t_4, t_4)\}$ there exists a homomorphism $h : \tilde{T} \to T$ such that $h(p) = t_i$ and $h(q) = t_j$.*

**Proof.** Suppose that there exists $h : \tilde{T} \to T$ such that $h(p) = h(q)$. Since $\tilde{S}_{21}$ and $\tilde{S}_{34}$ are extended choosers, then either $h(a_1^1) = h(a_2^1) = h(a_3^1)$ or $h(a_1^2) = h(a_2^2) = h(a_3^2)$, which implies a contradiction with Claim 5.1.13. For the second part of the claim, consider the pair

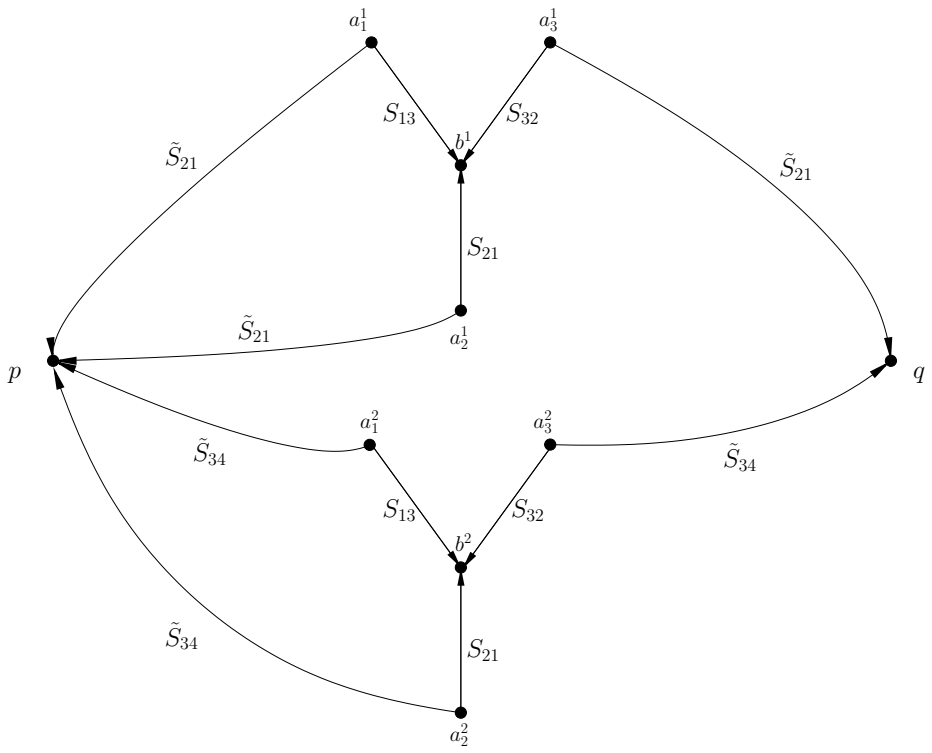


Figure 5.13: The gadget $\tilde{T}$.

$(t_1, t_2)$. We shall define $h : \tilde{T} \to T$ such that $h(p) = t_1$ and $h(q) = t_2$. Define, of course, $h(p)$ and $h(q)$ to be $t_1$ and $t_2$. Now, define $h(a_1^1) = t_1, h(a_2^1) = t_1$ and $h(a_3^1) = t_2$. We can extend $h$ to the copies of $\tilde{S}_{21}$, using the definition of extended chooser. Also we can extend $h$ to $T_1'$ using Claim 5.1.13. On the other hand, for any image that we choose for $a_1^2, a_2^2$ and $a_3^2$, we can extend $h$ to the copies of $\tilde{S}_{34}$, so we choose suitable images for them so that we can use Claim 5.1.13 and extend $h$ to $T_2'$. The case $(t_2, t_1)$ is symmetric and the case $(t_3, t_4)$ is analogous. For the other cases is easier to choose the images for the $a_j^i$'s, so we have the result. □

From Claim 5.1.14 and 5.1.12, we have the following:

**Corollary 5.1.15.** *For any pair $(t_i, t_j) \in \{t_1, t_2, t_3\}^2 \setminus \{(t_1, t_1), (t_2, t_2), (t_3, t_3)\}$ there exists a homomorphism $h : \tilde{T} \to T$ such that $h(p) = t_i$ and $h(q) = t_j$ and the image of $h$ does not have common vertices with the subgraph $T_4 \cdot T_5^{-1} - v$ of $T$.*

Now we prove the DP-completeness of the weaker version of the Acyclic Approximation Problem:

**Proof of Proposition 5.1.2.** The membership in DP is clear. To prove the hardness, consider the Exact-Four-Colorabiliy Problem: Given an (undirected) graph $G$, decide if $G$ is

4-colorable but not 3-colorable. It is known, see [32], that this problem is DP-complete. We shall define a polynomial reduction $f$ of the Exact-Four-Colorability Problem to the weaker version of the Acyclic Approximation Problem, concluding the result.

Given a graph $G$ define the digraph $\varphi(G)$ as follows: Consider $V(G)$ as the vertices of $\varphi(G)$. For each edge $\{u, u'\} \in E(G)$, add a new disjoint copy of $\tilde{T}$ and identify $p$ with $u$ and $q$ with $u'$. Add a new vertex $v_0$. For each vertex $u \in V(G) \subseteq V(\varphi(G))$ add new disjoint copies of $Q^*$ and $T_5$, and identify the initial vertex of the copy of $Q^*$ with $v_0$, the terminal vertex of the copy of $Q^*$ with $u$ and the terminal vertex of the copy of $T_5$ with $u$, see Figure 5.14. Finally, define $f(G) = (\varphi(G), T)$.

Clearly, $f$ can be computed in polynomial time in $|G|$. It suffices to prove that $G$ is 4-colorable but not 3-colorable if and only if $\varphi(G) \to T$ but there is no proper subgraph $S$ of $T$ such that $\varphi(G) \to S$ (note that, since $T$ is a core, that there is no proper subgraph $S$ of $T$ with $\varphi(G) \to S < T$ if and only if there is no proper subgraph $S$ of $T$ such that $\varphi(G) \to S$).

Suppose $G$ is 4-colorable but not 3-colorable. Since, $G$ is 4-colorable, then there exists a 4-coloring $c : V(G) \to \{1, 2, 3, 4\}$ of $G$. We shall define a homomorphism $h : \varphi(G) \to T$. Define for each $u \in V(G) \subseteq V(\varphi(G))$, $h(u) = t_{c(u)}$. For each $\{u, u'\} \in E(G)$, it holds $h(u) \neq h(u')$, since $c$ is a coloring. Using Claim 5.1.14, we can extend $h$ to all the copies of $\tilde{T}$ in $\varphi(G)$. Finally, define $h(v_0) = v$. Now, the images of the copies of $Q^*$and $T_5$ in $\varphi(G)$ are completely determined. For example, if $u \in V(G) \subseteq V(\varphi(G))$ is such that $h(u) = t_1$, then the copy of $Q^*$ associated with $u$ has to be mapped to $T_1$ in $T$ and the copy of $T_5$ associated with $u$ has to be mapped to the copy of $T_5$ in $T$ whose initial vertex is $u_1$. Thus, $\varphi(G) \to T$.

By contradiction, suppose there exists a proper subgraph $S$ of $T$ such that $\varphi(G) \xrightarrow{g} S$. Then, necessarily exists $i^* \in \{1, 2, 3, 4\}$, such that $g(u) \neq t_{i^*}$ for all $u \in V(G) \subseteq V(\varphi(G))$. Suppose not, then for all $i \in \{1, 2, 3, 4\}$ there exists $u \in V(G) \subseteq V(\varphi(G))$ with $g(u) = t_i$. Consider $i = 1$ and take $u$ with $g(u) = t_1$. Since $hg(\varphi(G)) = 25$, then $g$ preserves levels, implying that $g(v_0)$ is either $v, u_1, u_2, u_3$ or $u_4$. Because $Q^*$ and $T_5$ are incomparable, it follows that $g(v_0) = v$. This implies that the copy of $Q^*$ associated to $u$ is mapped via $g$ to the copy of $T_1$ in $T$. Using Lemma 5.1.5, $Q^*$ is mapped via $g$ in a surjective manner. Also, using again the incomparability between $Q^*$ and $T_5$, it follows that the copy of $T_5$ associated with $u$ is mapped via $g$ to the copy of $T_5$ in $T$ whose initial vertex is $u_1$. Obviously, since $T_5$ is a core, this mapping is surjective as well. Then we conclude that $T_1 \cdot T_5^{-1}$ is contained in the homomorphic image of $g$. Since we can do this for all $i \in \{1, 2, 3, 4\}$, it follows that $g$ is surjective, which is a contradiction. Thus, effectively there exists $i^* \in \{1, 2, 3, 4\}$, such that $g(u) \neq t_{i^*}$ for all $u \in V(G) \subseteq V(\varphi(G))$. Using Claim 5.1.14 we have that $g(u) \neq g(u')$ for all $\{u, u'\} \in E(G)$. This clearly implies that $G$ is 3-colorable, which is a contradiction.

Now, suppose that $\varphi(G) \to T$ but there is no proper subgraph $S$ of $T$ such that $\varphi(G) \to S$.

Since $\varphi(G) \xrightarrow{h} T$, using Claim 5.1.14, it holds that $h(u) \neq h(u')$ for all $\{u, u'\} \in E(G)$ and we can define a 4-coloring for $G$. Indeed, just take $c : V(G) \to \{1, 2, 3, 4\}$ with $c(u) = i$ if $h(u) = t_i$.

By contradiction, suppose there exists a 3-coloring $c : V(G) \to \{1, 2, 3\}$ for $G$. We define a homomorphism $g$ as follows: For each $u \in V(G) \subseteq V(\varphi(G))$, let $g(u) = t_{c(u)}$. Since $c$ is a coloring, then $g(u) \neq g(u')$, for all $\{u, u'\} \in E(G)$. Using Corollary 5.1.15, it follows that we can extend $g$ to all $\varphi(G)$, in a way that $T_4 \cdot T_5^{-1}$ is not contained in the homomorphic image of $g$. Thus, $g$ is not surjective, which is a contradiction. □

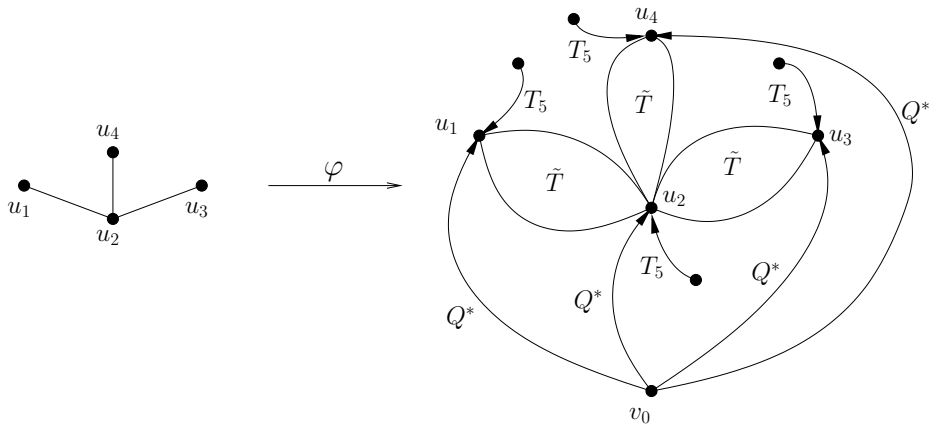


Figure 5.14: The digraph $\varphi(G)$ for the graph $G = (\{u_1, u_2, u_3, u_4\}, \{\{u_1, u_2\}, \{u_2, u_3\}, \{u_2, u_4\}\})$.

Now, we have the following:

**Proposition 5.1.16.** *Let $G$ be a graph. Then $\varphi(G) \to T$ but there is no proper subgraph $S$ of $T$ such that $\varphi(G) \to S$ if and only if $\varphi(G) \to T$ but there is no acyclic digraph $S$ such that $\varphi(G) \to S < T$.*

**Proof.** The backward direction is trivial. For the forward direction, suppose by contradiction that there exists an acyclic digraph $A$ such that $\varphi(G) \xrightarrow{h} A \xrightarrow{g} T$ and $T \not\to A$. Suppose that there exists $i^* \in \{1, 2, 3, 4\}$, such that $g \circ h(u) \neq t_{i^*}$ for all $u \in V(G) \subseteq V(\varphi(G))$. Using the same arguments as in proof of Proposition 5.1.2, we have that $G$ is 3-colorable and then there exists a proper subgraph $S$ of $T$ such that $\varphi(G) \to S$, which is a contradiction. Thus, necessarily, for all $i \in \{1, 2, 3, 4\}$ there exists $u \in V(G) \subseteq V(\varphi(G))$ such that $g \circ h(u) = t_i$. Consider $i = 1$ and take $u$ with $g \circ h(u) = t_1$. Let $Q_1^*$ be the copy of $Q^*$ associated with $u$ and let $A'$ be the homomorphic image via $h$ of $Q_1^*$. Using the same arguments as in the proof of Proposition 5.1.2 ($g \circ h$ preserves levels and the incomparability of $Q^*$ and $T_5$), it follows that

the homomorphic image of $Q_1^*$ via $g \circ h$ is exactly $T_1$. This implies that the homomorphic image of $A'$ via $g$ is $T_1$. Thus $Q_1^* \xrightarrow{h} A' \xrightarrow{g} T_1$.

Using the fact that $A'$ is acyclic (is a subgraph of $A$) and Claim 5.1.6, we have that there exists a homomorphism $g' : T_1 \to A'$. Let $h_5$ be the restriction of $h$ to the copy of $T_5$ associated to $u$. We shall define a homomorphism $r_1$ from $T_1 \cdot T_5^{-1} \subseteq T$ to $A$. For each $z$ in the copy of $T_1$, define $r_1(z) = g'(z)$ and for each $z$ in the copy of $T_5$ define $r_1(z) = h_5(z)$. We shall see that $r_1$ is well defined, i.e., $g'(t_1) = h_5(t_1)$. Using Lemma 4.2.3, we know that $hg(A) = 25$, so $h$ preserves levels. Thus, $h(u)$ and $h(v_0)$ are the only vertex in $A'$ with level 25 and 0, respectively. Again, since $hg(A') = 25$ as well, then $g'$ preserves levels, implying that $g'(t_1) = h(u) = h_5(u) = h_5(t_1)$ and $g'(v) = h(v_0)$. Thus, $r_1$ is well defined, clearly is a homomorphism and $r_1(v) = g'(v) = h(v_0)$. We can do this for all $i \in \{1, 2, 3, 4\}$, obtaining for each $1 \leq i \leq 4$ a homomorphism $r_i$ from $T_i \cdot T_5^{-1} \subseteq T$ to $A$, such that $r_i(v) = h(v_0)$. Finally, we define a homomorphism $r : T \to A$ such that $r(u) = r_i(u)$ if $u \in V(T_i \cdot T_5^{-1}) \subseteq V(T)$. Since $v$ is the only common vertex of the subgraphs $T_i \cdot T_5^{-1}$'s and $r_1(v) = r_2(v) = r_3(v) = r_4(v) = h(v_0)$, $r$ is well defined and is a homomorphism. This is a contradiction. □

Finally, using the same reduction $f$ as in the proof of Proposition 5.1.2 and the Proposition 5.1.16, the DP-completeness of the Acyclic Approximation Problem follows easily.

Now, we shall show that the Acyclic Approximation Problem is DP-complete even when the tableau of $Q$ and $Q'$ are cores and $Q'$ is fixed. We shall define a new function $\tilde{\varphi}(\cdot)$ from $\varphi(\cdot)$, such that for all undirected $G$, $\tilde{\varphi}(G)$ is a core and $G$ is 4-colorable and no 3-colorable if and only if $\tilde{\varphi}(G) \to T$ and there is no acyclic $S$ such that $\tilde{\varphi}(G) \to S < T$. Since $T$ is already a core and is fixed in the reduction, this will prove the result.

First, note that $\tilde{T}$ is not a core, since for each $a_j^i$ there are two disjoint copies of $T_{12}$ whose terminal vertices are identified with $a_j^i$. Thus, we modify $\tilde{T}$, leaving only one copy of those $T_{12}$ for each $a_j^i$. Observe that Claim 5.1.14 is still valid. We have the following:

**Claim 5.1.17.** *If $h$ is a homomorphism from $\tilde{T}$ to $\tilde{T}$ such that $h(p) = p$ and $h(q) = q$, then $h$ is the identity mapping.*

**Proof.** This follows easily using Lemma 4.2.4, Claim 5.1.7, Claim 5.1.8 and the fact that $T_X \not\to T_Y$, where $T_X$ is an oriented tree from Claim 5.1.7 or 5.1.8, with $X$ denoting the set of indices and $Y \not\subseteq X$ (otherwise, by composition, we would have $T_X \to T_k$ for some $k \notin X$). □

Now we shall define some digraphs. For $n \geq 1$, consider the oriented path $W_n = 000(10)^n0$, see Figure 5.15. Note that $hg(W_n) = 4$. Now, for each $1 \leq k \leq n$, define $W_n^k$ to

be $W_n$ plus an edge from a new element $z_k$ to $x_k$, see Figure 5.16. Note that $hg(W_n^k) = 4$ as well. We have the following:

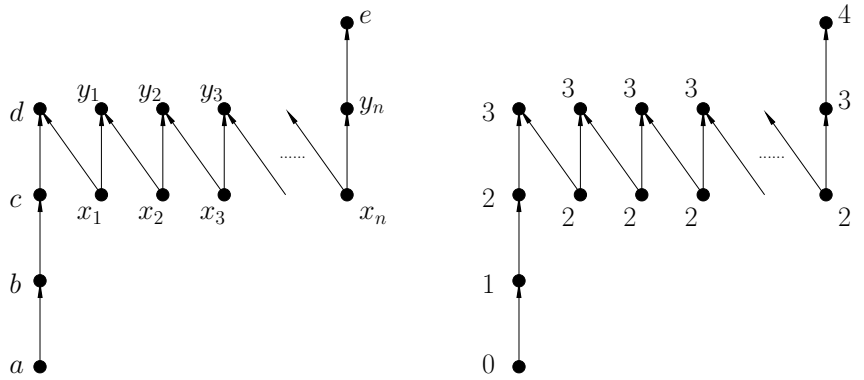


Figure 5.15: The digraph $W_n$ and its levels.

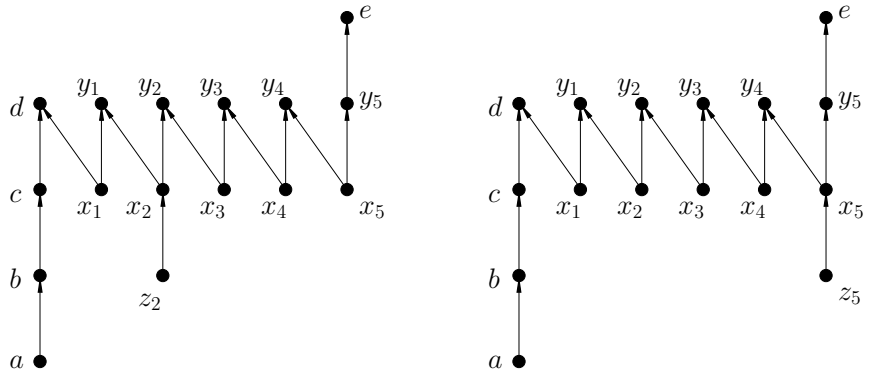


Figure 5.16: The digraphs $W_5^2$ and $W_5^5$.

**Claim 5.1.18.** *For each $n \geq 1$, the digraphs $W_n^k$, with $k \in \{1, ..., n\}$, are incomparable cores.*

**Proof.** Suppose that $W_n^k$ is not a core for some $1 \leq k \leq n$. Then there exists $W_n^k \xrightarrow{h} W_n^k$, where $h$ is not surjective. Necessarily, $h(a) = a$, $h(b) = b$, $h(c) = c$, $h(d) = d$ and $h(e) = e$ (since $h$ preserves levels). This implies that $h(x_i) = x_i$ and $h(y_i) = y_i$ for each $1 \leq i \leq n$, see Figure 5.16. Since $h(x_k) = x_k$, necessarily $h(z_k) = z_k$. This implies that $h$ is surjective, which is a contradiction.

Now, suppose that $W_n^k \xrightarrow{h} W_n^{k'}$ for $k \neq k'$. Since $h$ preserves levels, we have that $h$ maps $a, b, c, d$ and $e$ in $W_n^k$ to $a, b, c, d$ and $e$ in $W_n^{k'}$, respectively. This implies that $h$ maps for each $1 \leq i \leq n$, $x_i$ and $y_i$ in $W_n^k$ to $x_i$ and $y_i$ in $W_n^{k'}$, respectively. Since $k \neq k'$, we have that $z_k$ cannot be mapped in $W_n^{k'}$ via $h$, which is a contradiction. □

Consider the digraph $S$ as defined in Figure 5.17. The oriented paths $P_6$ and $P_8$ are as defined in the beginning of the proof and $P_{135}$ is from Claim 5.1.4.

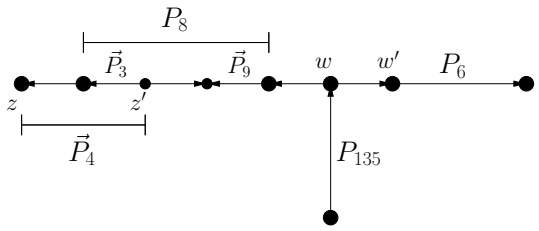


Figure 5.17: The digraph $S$.

Now, for each $n \geq 1$ and $1 \leq k \leq n$, we define the digraph $S_n^k$ as follows: Consider $S$ and replace the directed path of length 4 in $S$ that starts at $z'$ and ends at $z$ by a copy of $W_n^k$ identifying $a$ with $z'$ and renaming $e$ to $z$, see Figure 5.18. Observe that $W_n^k \to \vec{P}_4$, thus $S_n^k \to S$.

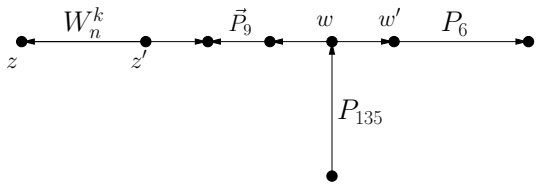


Figure 5.18: The digraph $S_n^k$.

We have the following:

**Claim 5.1.19.** *For each $n \geq 1$, the digraphs $S_n^k$, with $k \in \{1, ..., n\}$, are incomparable cores.*

**Proof.** Follows easily from Lemma 4.2.4, Claim 5.1.18 and the fact that $P_6$ and $P_8$ are incomparable. □

Now we define our main construction. Consider an undirected graph $G$ and let $u_1$,$u_2$,...,$u_n$ be its vertices. We define $\tilde{\varphi}(G)$ to be $\varphi(G)$ but in addition, for each $1 \leq k \leq n$ we add a disjoint copy of $S_n^k$ and identify $z$ in $S_n^k$ with $u_k$. Clearly, $\tilde{\varphi}(G)$ can be computed in polynomial time in $|G|$. To conclude, we shall prove the following proposition:

**Proposition 5.1.20.** *For all graph $G$, the digraph $\tilde{\varphi}(G)$ is a core. Furthermore, $G$ is 4-colorable and no 3-colorable if and only if $\tilde{\varphi}(G) \to T$ and there is no acyclic $S$ such that $\tilde{\varphi}(G) \to S < T$.*

**Proof.** Let $G$ be a graph and $h$ be a homomorphism from $\tilde{\varphi}(G)$ to $\tilde{\varphi}(G)$. We shall prove that $h$ is surjective, implying that $\tilde{\varphi}(G)$ is a core. Recall that $u_1, ..., u_n$, the vertices of $G$, are contained in $\tilde{\varphi}(G)$ as well. We have that $h(u_k) = u_k$, for each $1 \leq k \leq n$. Indeed, since $h$ preserves levels, we know that $h$ maps $u_k$ to $u_l$, for some $1 \leq l \leq n$, or maps $u_k$ to some vertex in a copy of $\tilde{T}$ with level 25. Using the facts that $Q^* \not\to T_X$, with $5 \in X$ (since $Q^* \not\to T_5$) and $T_X \not\to T_Y$, with $Y \not\subseteq X$, we can easily show that the second case is not possible, since we have copies of $T_{345}$ (from $\tilde{S}_{21}$), $T_{15}$ (from $\tilde{S}_{34}$) and $Q^*$ whose terminal vertices are identified with $u_k$, see Figure 5.19. For example, $h$ cannot map $u_k$ to $a_1^1$, otherwise $T_{345} \to T_{12}$, $T_{345} \to T_{15}$ or $T_{345} \to T_{125}$. Similarly, $h$ cannot map $u_k$ to $b^1$, otherwise $Q^* \to T_{15}$, $Q^* \to T_{25}$ or $Q^* \to T_{35}$, see Figure 5.19. For the others vertices in $\tilde{T}$ with level 25, we have a similar contradictions.

Thus, $h$ maps $u_k$ to $u_l$, for some $1 \leq l \leq n$. We shall show that $l = k$. Indeed, suppose that $l \neq k$. Since the only vertex with level 25 in the copy of $S_n^k$ whose terminal vertex is $u_k$, is precisely $u_k$, we have that $h$ maps this copy of $S_n^k$ either to a copy of $Q^*$, $T_{345}$, $T_{15}$, $T_5$ or $S_n^l$, whose terminal vertex is $u_l$. Suppose that $h$ maps the copy of $S_n^k$ to the copy of $Q^*$. Necessarily (using Lemma 4.2.4), $h$ maps $w$ in $S_n^k$ to $a_7$ in $Q^*$, see Figures 5.18 and 5.1. This would imply that $P_{135} \to P_7$, which is a contradiction with Claim 5.1.4. Now, suppose that $h$ maps the copy of $S_n^k$ to the copy of $T_5$. It follows that $h$ maps $w'$ in $S_n^k$ to the initial vertex of the copy of $P_8$ in $T_5$, implying that $P_6 \to P_8$ or $P_6 \to P_9$, which is a contradiction too. The cases $T_{345}$ and $T_{15}$ lead to a contradiction as well, since $T_{345} \to T_5$ a nd $T_{15} \to T_5$. Finally, suppose that $h$ maps the copy of $S_n^k$ to the copy of $S_n^l$ whose terminal vertex is $u_l$. It follows that $S_n^k \to S_n^l$, which contradicts Claim 5.1.19. In any case we have a contradiction, thus $l = k$ and $h(u_k) = u_k$, for each $1 \leq k \leq n$.

Now, observe that for each $1 \leq k \leq n$, the copy of $T_5$ and $S_n^k$ whose terminal vertex is $u_k$ have to be mapped via $h$ to themselves. Moreover, $h(v_0) = v_0$, otherwise $Q^* \to T_5$, $Q^* \to T_{15}$ or $Q^* \to T_{345}$, which is impossible. This implies that $h$ maps all the copies of $Q^*$ to themselves. Finally, observe that $h$ maps all the copies of $\tilde{T}$ to themselves as well. Indeed, using the facts that $h$ preserves levels, $T_X \not\to T_Y$, with $Y \not\subseteq X$ and $h(u_k) = u_k$, for each $1 \leq k \leq n$, we can easily show that $h$ maps each vertex in a copy of $\tilde{T}$ to a vertex inside the same copy of $\tilde{T}$. Thus, we can use Claim 5.1.17, to conclude that $h$ maps all the copies of $\tilde{T}$ to themselves. This prove that $\tilde{\varphi}(G)$ is a core.

Finally, observe that for each $1 \leq i \leq 4$ and $1 \leq k \leq n$, there is a homomorphism $g_i$ from $S_n^k$ to $T_i$ such that $g_i$ maps the vertex $z$ in $S_n^k$ to the terminal vertex of $T_i$. Thus, if we are constructing a homomorphism from $\tilde{\varphi}(G)$ to $T$, for any values of the images of the $u_k$'s, we always can define correct images for the copies of the $S_n^k$'s. Therefore, we can use exactly the same arguments as in proof of Proposition 5.1.2 and 5.1.16. □

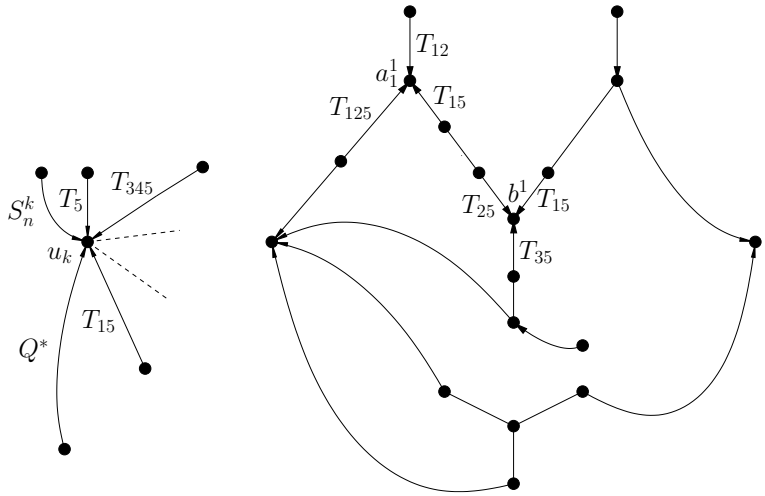


Figure 5.19: The vertex $u_k$ and a copy of $\tilde{T}$.

This proves that the Acyclic Approximation Problem is DP-complete even when the tableau of $Q$ and $Q'$ are cores and $Q'$ is fixed.

## 5.2 Bounded treewidth approximations

Observe from Corollary 4.4.2 that if a Boolean CQ $Q$ has a nontrivial $\mathsf{TW}(k)$-approximation, then the query $Q^{\text{triv}}_{k+1}$ with the tableau $K^{\rightleftarrows}_{k+1}$ is contained in $Q$. For $k = 1$, we had a necessary and sufficient condition for such a query to be an approximation (it was the PTIME-testable condition of not being balanced, see Theorem 4.3.1). For $\mathsf{TW}(k)$, we do not have such a characterization, but we do know that even for $\mathsf{TW}(2)$, the criterion will be much harder than for the acyclic case due to the following.

**Proposition 5.2.1.** *For every $k > 1$, testing, for a Boolean CQ $Q$ over graphs, whether $Q^{\text{triv}}_{k+1}$ is a $\mathsf{TW}(k)$-approximation of $Q$ is* NP*-hard.*

**Proof.** Let $k > 1$. We shall prove that there is a polynomial reduction from the $(k+1)$-coloring problem to our problem.

Let $G$ be a graph. The reduction returns the CQ $\varphi(G)$, whose tableau is $\vec{G} + K^{\rightleftarrows}_{k+1}$ (where $+$ denotes disjoint union), where $\vec{G}$ is the directed version of $G$ (replace $\{a, b\}$ by $(a, b)$ and $(b, a)$). It suffices to show that $G$ is $(k+1)$-colorable iff $Q^{\text{triv}}_{k+1}$ is a $\mathsf{TW}(k)$-approximation of $\varphi(G)$.

Suppose that $G$ is $(k+1)$-colorable, then $\vec{G} \to K^{\rightleftarrows}_{k+1}$, implying that the tableau of $\varphi(G)$ is homomorphically equivalent to $K^{\rightleftarrows}_{k+1}$, the tableau of $Q^{\text{triv}}_{k+1}$. Therefore, $\varphi(G)$ is equivalent to $Q^{\text{triv}}_{k+1}$ and thus $Q^{\text{triv}}_{k+1}$ is a $\mathsf{TW}(k)$-approximation of $\varphi(G)$.

Now, suppose that $Q^{\text{triv}}_{k+1}$ is a $\mathsf{TW}(k)$-approximation of $\varphi(G)$. In particular, we have that $T_{\varphi(G)} \to T_{Q^{\text{triv}}_{k+1}}$, implying that $\vec{G} \to K^{\rightleftarrows}_{k+1}$.Therefore, $G$ is $(k+1)$-colorable. □

Thus, while the behavior of acyclic and treewidth-$k$ approximations for $k > 1$ is in general similar, testing conditions that guarantee certain properties of approximations is harder even for treewidth-2, compared to the acyclic case.

Finally, we note that the analog of the Acyclic Approximation problem for treewidth $k$ (i.e., checking if $Q'$ is a $\mathsf{TW}(k)$-approximation of $Q$) remains DP-complete for all $k \geq 1$. Indeed, the proof of the upper bound for the acyclic case applies to bounded treewidth, and the lower bound is already established for $k = 1$.

# Chapter 6

# Extensions

## 6.1 Approximating arbitrary queries

We now switch to queries over arbitrary vocabularies and extend some previous results. We omit all proofs as techniques used here are similar than those used in previous chapters. For complete proofs we refer the reader to [36].

For arbitrary queries, tractability restrictions could be either *graph-based* and *hypergraph-based*. For the graph-based notions, one deals with the graph of query $Q$, denoted by $G(Q)$. The nodes of $G(Q)$ are variables used in $Q$. If there is an atom $R(x_1, \ldots, x_n)$ in $Q$, then $G(Q)$ has undirected edges $\{x_i, x_j\}$ for all $1 \leq i < j \leq n$. Note that for graph queries, we have $G(Q) = T_Q^{\mathrm{u}}$.

For hypergraph-based notions, we put restrictions on the hypergraph $\mathcal{H}(Q)$ of $Q$. Recall that its nodes again are variables used in $Q$, and its hyperedges correspond to the atoms of $Q$, i.e., for each atom $R(x_1, \ldots, x_n)$ in $Q$, we have a hyperedge $\{x_1, \ldots, x_n\}$.

Restrictions on queries are imposed as follows. If $\mathcal{C}$ is a class of graphs, or hypergraphs, then a query $Q$ is

- a *graph-based* $\mathcal{C}$*-query* if $G(Q) \in \mathcal{C}$, and
- a *hypergraph-based* $\mathcal{C}$*-query* if $\mathcal{H}(Q) \in \mathcal{C}$.

In general, these are incompatible: there are graph-based classes that are not hypergraph-based, and vice versa [12].

### 6.1.1 Graph-based classes

For graph-based queries, it is easy to transfer results from queries over graphs to queries over arbitrary schemas. We state the result below only for the classes of tractable graph-based

queries, but a general existence theorem, extending Theorem 4.0.1, is true as well.

Tractability of CQ answering with respect to graph-based classes of queries was fully characterized in [17]: given a class $\mathcal{C}$, query answering for graph-based $\mathcal{C}$-queries is tractable iff $\mathcal{C} \subseteq \mathsf{TW}(k)$ for some $k$.

We call a CQ $Q'$ a *graph-based $\mathcal{C}$-approximation* of $Q$ if it is an approximation of $Q$ in the class of graph-based $\mathcal{C}$-queries. Then we have an analog of the existence of approximation results from Chapter 4.

**Theorem 6.1.1.** *Every CQ $Q$ has a graph-based $\mathsf{TW}(k)$-approximation, for every $k \geq 1$, with at most as many joins as $Q$. Moreover, such an approximation can be found in single-exponential time.*

### 6.1.2 Hypergraph-based classes

We now look at *hypergraph-based $\mathcal{C}$-approximations*, i.e., approximations in the class of hypergraph-based $\mathcal{C}$ queries.

The oldest tractability criterion for CQs, acyclicity [34], is a hypergraph-based notion (see the definition in Chapter 3). An analog of bounded treewidth for hypergraphs was defined in [16]; that notion of bounded *hypertree width* properly extended acyclicity and led to tractable classes of CQs over arbitrary vocabularies.

Our first goal, therefore, is to have a general result about the existence of approximations that will apply to both acyclicity and bounded hypertree width (to be defined formally shortly).

Note we cannot trivially lift the closure condition used in Theorem 4.0.1 for hypergraphs, since even acyclic hypergraphs are not closed under taking subhypergraphs. Indeed, take a hypergraph $\mathcal{H}$ with hyperedges $\{a, b, c\}, \{a, b\}, \{b, c\}, \{a, c\}$. It is acyclic: the decomposition has $\{a, b, c\}$ associated with the root of the tree, and two-element edges with the children of the root. However, it has cyclic subhypergraphs, for instance, one that contains its two-element edges.

The closure conditions we use instead are:

- *Closure under induced subhypergraphs.* If $\mathcal{H} = \langle V, \mathcal{E}\rangle$ is in $\mathcal{C}$ and $\mathcal{H}'$ is an induced subhypergraph, then $\mathcal{H}' \in \mathcal{C}$. Recall that an induced subhypergraph is one of the form $\langle V', \{e \cap V' \mid e \in \mathcal{E}\}\rangle$.
- *Closure under edge extensions:* if $\mathcal{H} = \langle V, \mathcal{E}\rangle$ is in $\mathcal{C}$ and $\mathcal{H}'$ is obtained by adding new vertices $V'$ to one hyperedge $e \in \mathcal{E}$, where $V'$ is disjoint from $V$, then $\mathcal{H}' \in \mathcal{C}$.

We shall see that these will be satisfied by the classes of hypergraphs of interest to us. The analog of the previous existence results can now be stated as follows.

**Theorem 6.1.2.** *Let $\mathcal{C}$ be a class of hypergraphs closed under induced subhypergraphs and edge extensions. Then every CQ $Q$ that has at least one hypergraph-based $\mathcal{C}$-query contained in it, has a hypergraph-based $\mathcal{C}$-approximation.*

*Moreover, the number of non-equivalent hypergraph based $\mathcal{C}$-approximations of $Q$ is at most exponential in the size of $Q$, and every such approximation is equivalent to one which has at most $O(n^{m-1})$ variables and at most $O(n^m)$ joins, where $n$ is the number of variables in $Q$, and $m$ is the maximum arity of a relation in the vocabulary.*

It is straightforward to check that the class of acyclic hypergraphs satisfies both closure conditions, and that any constant homomorphism on a query $Q$ produces an acyclic query. Thus,

**Corollary 6.1.3.** *For every vocabulary $\sigma$, there exist two polynomials $p_\sigma$ and $r_\sigma$ such that every CQ $Q$ over $\sigma$ has a hypergraph-based acyclic approximation of size at most $p_\sigma(|Q|)$ that can be found in time $2^{r_\sigma(|Q|)}$.*

Next, we extend these results to hypertree width. First we recall the definitions [16]. A *hypertree decomposition* of a hypergraph $\mathcal{H} = \langle V, \mathcal{E}\rangle$ is a triple $\langle T, f, c\rangle$, where $T$ is a rooted tree, $f$ is a map from $T$ to $2^V$ and $c$ is a map from $T$ to $2^{\mathcal{E}}$, such that

- $(T, f)$ is a tree decomposition of $\mathcal{H}$.
- $f(u) \subseteq \bigcup c(u)$ holds for every $u \in T$.
- $\bigcup c(u) \cap \bigcup\{f(t) \mid t \in T_u\} \subseteq f(u)$ holds for every $u \in T$, where $T_u$ refers to the subtree of $T$ rooted at $u$.

The *width* of a hypertree decomposition $\langle T, f, c\rangle$ is $\max_{u\in T} |c(u)|$. The *hypertree width* $hw(\mathcal{H})$ of $\mathcal{H}$ is the minimum width over all its hypertree decompositions. We denote by $\mathsf{HTW}(k)$ the class of hypergraphs with hypertree width at most $k$, and slightly abusing notation, the class of CQ's or tableaux whose hypergraphs have hypertree width at most $k$.

The key result of [16] is that for each fixed $k$, CQs from $\mathsf{HTW}(k)$ can be evaluated in polynomial time with respect to combined complexity. It is also shown in [16] that a hypergraph $\mathcal{H}$ is acyclic iff its hypertree width is 1. That is, $\mathsf{AC} = \mathsf{HTW}(1)$.

To apply the existence result, we need to check the closure conditions for hypergraphs of fixed hypertree width. It turns out they are satisfied.

**Lemma 6.1.4.** *For each $k$, the class $\mathsf{HTW}(k)$ is closed under induced subhypergraphs and edge extensions.*

This gives us the desired result about the existence of approximations within $\mathsf{HTW}(k)$ for every $k$.

**Corollary 6.1.5.** *For every vocabulary $\sigma$, there exist two polynomials $p_\sigma$ and $r_\sigma$ such that every CQ $Q$ over $\sigma$ has a hypergraph-based $\mathsf{HTW}(k)$-approximation of size at most $p_\sigma(|Q|)$ that can be found in time $2^{r_\sigma(|Q|)}$, for every $k \geq 1$.*

We finish with an example that illustrate the richness of approximations when we go from graphs queries to arbitrary queries.

**Example 6.1.6.** Consider a Boolean query

$$Q() \coloneq R(x_1, x_2, x_3), R(x_3, x_4, x_5), R(x_5, x_6, x_1)$$

over a schema with one ternary relation. If we had a binary relation instead and omitted the middle attribute, we would obtain a query whose tableau is a cycle of length 3, thus having only trivial approximations. However, going beyond graphs lets us find nontrivial acyclic approximations. In fact this query has at least 3 non-equivalent acyclic approximations (all queries below are minimized):

- With fewer joins than $Q$:
$$Q'_1() \coloneq R(x, y, x).$$
- With as many joins as $Q$:
$$Q'_2() \coloneq R(x_1, x_2, x_3), R(x_3, x_4, x_2), R(x_2, x_5, x_1).$$
- With more joins than $Q$:
$$\begin{aligned} Q'_3() \quad \coloneq \quad & R(x_1, x_2, x_3), R(x_3, x_4, x_5), \\ & R(x_5, x_6, x_1), R(x_1, x_3, x_5). \end{aligned}$$

## 6.2 Overapproximations

So far approximations were required to produce only correct answers, i.e., no false positives. In general, one can change and/or relax this assumption. One obvious way is to look for approximations that produce all correct answers, and perhaps something else, i.e., no false negatives. We refer to them as *overapproximations*. For complete proofs of this chapter see [36].

Formally, for a CQ $Q$ not in a class $\mathcal{C}$ of CQs, a query $Q' \in \mathcal{C}$ such that $Q \subseteq Q'$ is a *$\mathcal{C}$-overapproximation* of $Q$ if there is no query $Q'' \in \mathcal{C}$ with $Q \subseteq Q''$ such that $Q' \sqsubset_Q Q''$. In other words, $Q'$ is an overapproximation of $Q$ if it is guaranteed to return all results of $Q$ and no other such query approximates $Q$ better than $Q'$.

Recall that for acyclic approximations of Boolean graph queries, we had two cases which resulted in trivial approximations $Q^{\text{triv}}$ and $Q_2^{\text{triv}}$, whose tableaux are the single-element loop $K_1^{\circlearrowleft}$ and the graph $K_2^{\rightleftarrows}$. These may serve as acyclic overapproximations as well, but in general we have both existence and nonexistence results.

**Proposition 6.2.1.** *Let $Q$ be a cyclic Boolean CQ over graphs. Then:*

- *If $T_Q$ contains $K_1^{\circlearrowleft}$ as a subgraph, then $Q^{\text{triv}}$ is the unique, up to equivalence, acyclic overapproximation of $Q$.*
- *If $T_Q$ contains $K_2^{\rightleftarrows}$ as a subgraph, but does not contain $K_1^{\circlearrowleft}$, then $Q_2^{\text{triv}}$ is the unique, up to equivalence, acyclic overapproximation of $Q$.*
- *If $T_Q$ contains neither $K_1^{\circlearrowleft}$ nor $K_2^{\rightleftarrows}$ as a subgraph, then:*
  1. *When $T_Q$ is not a balanced graph, $Q$ has no acyclic overapproximations.*
  2. *When $T_Q$ is a balanced graph, $Q$ may or may not have acyclic overapproximations.*

Even though we do not yet have a criterion for the existence of acyclic approximations of $Q$ when the tableau of $Q$ is balanced, we can say a lot about their structure and size when they exist: they will be closely related to spanning forests of $T_Q$, and will be bounded by the size of $Q$.

**Theorem 6.2.2.** *If a cyclic Boolean CQ $Q$ has an acyclic overapproximation, then, up to equivalence, it has exactly one, whose tableau is homomorphically equivalent to a subgraph of $T_Q$ (in fact, to a spanning forest of $T_Q$).*

Hence, if an acyclic overapproximation exists, its minimization will have fewer joins than $Q$, because it will be the core of a subgraph of $T_Q$. Note that $T_Q$ may have multiple spanning forests, but only one of them (up to homomorphic equivalence) corresponds to the acyclic overapproximation.

Even more generally, one can define arbitrary approximations of $Q$ that do not impose any conditions on $Q'$ except it being maximal with respect to $\sqsubseteq_Q$. We leave their study to future work, and conclude by the analysis of complexity of the relation $\sqsubseteq_Q$.

**Proposition 6.2.3.** *The following problem is* NP*-complete: given three CQs $Q, Q_1, Q_2$, is $Q_1 \sqsubseteq_Q Q_2$? It remains* NP*-complete if $Q_1, Q_2$ are restricted to be from the class* $\mathsf{AC}$ *of acyclic queries, or from* $\mathsf{TW}(k)$ *for $k \geq 1$.*

Note that even the upper bound in this proposition is not straightforward, as the definition of $\sqsubseteq_Q$ involves universal quantification over databases, and then checking whether CQs evaluate to true or false over them. Even if we could prove that it suffices to consider

databases of at most polynomial size, just parsing the definition would have given us a $\Pi_2^p$ upper bound. The proof instead establishes structural properties of $\sqsubseteq_Q$ based on the properties of the $\rightarrow$ ordering.

# Chapter 7

# Conclusions

We have primarily concentrated on approximations of conjunctive queries that are guaranteed to return correct answers. Given the importance of acyclic CQs and very good complexity bounds for them, we have focused on acyclic approximations, but we also provided results on approximations within classes of bounded treewidth and bounded hypertree width. We have proved the existence of approximations, and showed they can be found with an acceptable computational overhead, and that their sizes are at most polynomial in the size of the original query, and sometimes are bounded by the size of the original query.

Here we dealt with the qualitative approach to approximations; in the future we would like to study quantitative guarantees as well, by defining measures showing how different from the original query approximations are. One approach is to find probabilistic guarantees for approximations. Note that such guarantees have been studied for queries from expressive languages (e.g., with fixed points or infinitary connectives) [1, 22], with typical results showing that queries are equivalent to those from simpler logics (e.g, FO) almost everywhere. One possibility is to specialize these results to much weaker logics, e.g., to CQs and their tractable subclasses.

# Bibliography


[1] S. Abiteboul, K. Compton, and V. Vianu. Queries are easier than you thought (probably). In *ACM Symp. on Principles of Database Systems*, 1992, ACM Press, pages 23–32.

[2] S. Abiteboul, O. Duschka. Complexity of answering queries using materialized views. In *PODS 1998*, pages 254–263.

[3] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[4] M. Aigner. *Combinatorial Theory*. Springer, 1997.

[5] H. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25 (1996), 1305–1317.

[6] B. ten Cate, Ph. Kolaitis, W.-C. Tan. Database constraints and homomorphism dualities. In *CP 2010*, pages 475–490.

[7] A. Chandra and P. Merlin. Optimal implementation of conjunctive queries in relational data bases. In *ACM Symp. on Theory of Computing*, 1977, pages 77–90.

[8] C. Chekuri, A. Rajaraman. Conjunctive query containment revisited. *Theor. Comput. Sci.* 239(2): 211-229 (2000).

[9] R. Fagin, Ph. Kolaitis, L. Popa. Data exchange: getting to the core. *ACM TODS* 30 (2005), 90–101.

[10] W. Fan, J. Li, S. Ma, N. Tang, Y. Wu. Graph pattern matching: from intractable to polynomial time. *PVLDB* 3(1): 264-275 (2010).

[11] R. Fink, D. Olteanu. On the optimal approximation of queries using tractable propositional languages. In *ICDT 2011*, pages 174–185.

[12] J. Flum, M. Frick, and M. Grohe. Query evaluation via tree-decompositions. *J. ACM*, 49 (2002), 716–752.

[13] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.

[14] M. Garofalakis and P. Gibbons. Approximate query processing: taming the terabytes. In *VLDB'01*.

[15] G. Gottlob, N. Leone, and F. Scarcello. The complexity of acyclic conjunctive queries. *J. ACM*, 48 (2001), 431–498.

[16] G. Gottlob, N. Leone, and F. Scarcello. Hypertree decompositions and tractable queries. *JCSS*, 64 (2002), 579–627.

[17] M. Grohe, T. Schwentick, and L. Segoufin. When is the evaluation of conjunctive queries tractable? In *ACM Symp. on Theory of Computing*, 2001, pages 657–666.

[18] W. Gutjahr, E. Welzl, G. Woeginger. Polynomial graph-colorings. *Discrete Applied Mathematics* 35(1): 29-45 (1992).

[19] A. Halevy. Answering queries using views: A survey. *VLDB J.* 10(4):270-294 (2001).

[20] P. Hell and J. Nešetřil. *Graphs and Homomorphisms*. Oxford University Press, 2004.

[21] P. Hell, J. Nešetřil, and X. Zhu. Complexity of tree homomorphisms. *Discrete Applied Mathematics* 70(1): 23-36 (1996).

[22] L. Hella, Ph. Kolaitis, and K. Luosto. Almost everywhere equivalence of logics in finite model theory. *Bull. of Symbolic Logic*, 2 (1996), 422–443.

[23] Y. Ioannidis. Approximations in database systems. In *ICDT'03*, pages 16–30.

[24] Ph. Kolaitis and M. Vardi. Conjunctive-query containment and constraint satisfaction. *JCSS* 61(2):302-332 (2000).

[25] Ph. Kolaitis and M. Vardi. A logical approach to constraint satisfaction. In *Finite Model Theory and Its Applications*, Springer 2007, pages 339–370.

[26] M. Lenzerini. Data integration: a theoretical perspective. In *PODS'02*, pages 233–246.

[27] L. Libkin. Incomplete information and certain answers in general data models. In *PODS 2011*, pages 59–70.

[28] Q. Liu. Approximate query processing. *Encyclopedia of Database Systems*, 2009, pages 113–119.

[29] S. Malik and L. Zhang. Boolean satisfiability: from theoretical hardness to practical success. *CACM* **52**(8), 76–82, 2009.

[30] C. H. Papadimitriou and M. Yannakakis. The complexity of facets (and some facets of complexity). *JCSS*, 28 (1986), 244–259.

[31] A. Robinson, A. Voronkov, eds. *Handbook of Automated Reasoning*. The MIT Press, 2001.

[32] J. Rothe. Exact complexity of exact-four-colorability. *Inf. Process. Lett.* 87(1): 7-12 (2003).

[33] M. Vardi. On the complexity of bounded-variable queries. In *PODS'95*, pages 266–276.

[34] M. Yannakakis. Algorithms for acyclic database schemes. In *Proc. Conf. on Very Large Databases*, 1981, pages 82–94.

[35] D. West. *Introduction to Graph Theory*. Prentice Hall, 2001.

[36] P. Barceló, L. Libkin and M. Romero. Efficient approximations of conjunctive queries. *Technical Report*. http://www.dcc.uchile.cl/∼mromero/publications/acq.pdf.