On the Complexity of Query Result Diversification

Citation for published version:
Deng, T & Fan, W 2014, 'On the Complexity of Query Result Diversification', *ACM Transactions on Database Systems*, vol. 39, no. 2, pp. 15. https://doi.org/10.1145/2602136

Digital Object Identifier (DOI):
10.1145/2602136

Published In:
ACM Transactions on Database Systems

On the Complexity of Query Result Diversification

TING DENG, RCBD and SKLSDE, Beihang University
WENFEI FAN, Informatics, University of Edinburgh, and RCBD and SKLSDE, Beihang University

Query result diversification is a bi-criteria optimization problem for ranking query results. Given a database D, a query Q and a positive integer k, it is to find a set of k tuples from $Q(D)$ such that the tuples are as relevant as possible to the query, and at the same time, as diverse as possible to each other. Subsets of $Q(D)$ are ranked by an objective function defined in terms of relevance and diversity. Query result diversification has found a variety of applications in databases, information retrieval and operations research.

This paper investigates the complexity of result diversification for relational queries. (1) We identify three problems in connection with query result diversification, to determine whether there exists a set of k tuples that is ranked above a bound with respect to relevance and diversity, to assess the rank of a given k-element set, and to count how many k-element sets are ranked above a given bound based on an objective function. (2) We study these problems for a variety of query languages and for the three objective functions proposed in [Gollapudi and Sharma 2009]. We establish the upper and lower bounds of these problems, *all matching*, for both combined complexity and data complexity. (3) We also investigate several special settings of these problems, identifying tractable cases. Moreover, (4) we re-investigate these problems in the presence of compatibility constraints commonly found in practice, and provide their complexity in all these settings.

Categories and Subject Descriptors: H.2.3 [**DATABASE MANAGEMENT**]: Languages

General Terms: Design, Algorithms, Theory

Additional Key Words and Phrases: Result diversification, relevance, diversity, recommender systems, database queries, combined complexity, data complexity, counting problems

1. INTRODUCTION

Result diversification for relational queries is a bi-criteria optimization problem. Given a query Q, a database D and a positive integer k, it is to find a set U of k tuples in the query result $Q(D)$ such that the tuples in U are as relevant as possible to query Q, and at the same time, as diverse as possible to each other. More specifically, we want to find a set $U \subseteq Q(D)$ such that $|U| = k$, and the value $F(U)$ of U is maximum. Here $F(\cdot)$ is called an *objective function*. It is defined on *sets* of tuples from $Q(D)$, in terms of a *relevance function* $\delta_{\mathsf{rel}}(\cdot,\cdot)$ and a *distance function* $\delta_{\mathsf{dis}}(\cdot,\cdot)$, where

— for each tuple $t \in Q(D)$, $\delta_{\mathsf{rel}}(t, Q)$ is a number indicating the relevance of answer t to query Q, such that the higher $\delta_{\mathsf{rel}}(t, Q)$ is, the more relevant t is to Q; and
— for all tuples $t_1, t_2 \in Q(D)$, $\delta_{\mathsf{dis}}(t_1, t_2)$ is the distance between t_1 and t_2, such that the larger $\delta_{\mathsf{dis}}(t_1, t_2)$ is, the more diverse the answers t_1 and t_2 are.

Deng and Fan are supported in part by 973 Program 2014CB340302. Fan is also supported in part by NSFC 61133002, 973 Program 2012CB316200, Guangdong Innovative Research Team Program 2011D005 and Shenzhen Peacock Program 1105100030834361, and EPSRC EP/J015377/1.
Author's addresses: T. Deng; W. Fan, School of Informatics, Laboratory for Foundations of Computer Science, Informatics Forum, 10 Crichton Street, Edinburgh EH8 9AB, Scotland, UK. Email: wenfei@inf.ed.ac.uk

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

In particular, three generic objective functions have been proposed and studied in [Gollapudi and Sharma 2009] based on an axiom system, namely, *max-sum diversification*, *max-min diversification* and *mono-objective formulation*. Each of these functions is defined in terms of generic functions $\delta_{\mathsf{rel}}(\cdot,\cdot)$ and $\delta_{\mathsf{dis}}(\cdot,\cdot)$, with a parameter $\lambda \in [0,1]$ specifying the tradeoff between relevance and diversity.

Query result diversification aims to improve user satisfaction by remedying the over-specification problem of retrieving too homogeneous answers. The diversity of query answers is measured in terms of (1) contents, to include items that are dissimilar to each other, (2) novelty, to retrieve items that contain new information not found in previous results, and (3) coverage, to cover items in different categories [Drosou and Pitoura 2010]. It has proven effective in Web search [Gollapudi and Sharma 2009; Vieira et al. 2011], recommender systems [Yu et al. 2009b; Zhang and Hurley 2008; Ziegler et al. 2005], databases [Demidova et al. 2010; Liu et al. 2009; Vee et al. 2008], sponsored-search advertising [Feuerstein et al. 2007], and in operations research and finance (see [Drosou and Pitoura 2010; Minack et al. 2009] for surveys).

This paper investigates the complexity of result diversification analysis for relational queries. While there has been a host of work on result diversification, the previous work has mostly focused on diversity and relevance metrics, and on algorithms for computing diverse results [Drosou and Pitoura 2010; Minack et al. 2009]. Few complexity results have been developed for query result diversification, and the known results are mostly lower bounds (NP-hardness) [Agrawal et al. 2009; Gollapudi and Sharma 2009; Liu et al. 2009; Stefanidis et al. 2010; Vieira et al. 2011]. Furthermore, these results are established by assuming that query result $Q(D)$ is *already known*. In other words, the prior work conducts diversification in two steps: first compute $Q(D)$, and then rank k-element subsets of $Q(D)$ and find a set with the maximum $F(\cdot)$ value. The known complexity results are for *the second step only*, based on *a specific objective function* $F(\cdot)$. However, it is typically expensive to compute $Q(D)$. To avoid the overhead, we want to combine the two steps by embedding diversification in query evaluation, and stop *as soon as* top-ranked results are found based on $F(\cdot)$ (*i.e.,* early termination), rather than to retrieve entire $Q(D)$ in advance [Demidova et al. 2010]. Nonetheless, the complexity of such a query result diversification process has not been studied.

This highlights the need for establishing the complexity of query result diversification, both upper bounds and lower bounds, when $Q(D)$ is *not provided*, and for *different query languages* and *various objective functions*. Indeed, to develop practical algorithms for computing diverse query results, we have to understand the impact of query languages and objective functions on the complexity of result diversification.

Example 1.1. Consider a recommender system to help people find gifts for various events or occasions, *e.g.,* FindGift[1]. Its underlying database D_0 consists of two relations specified by the following relation schemas:

$$\mathsf{catalog}(\mathsf{item}, \mathsf{type}, \mathsf{price}, \mathsf{inStock}),$$
$$\mathsf{history}(\mathsf{item}, \mathsf{buyer}, \mathsf{recipient}, \mathsf{gender}, \mathsf{age}, \mathsf{rel}, \mathsf{event}, \mathsf{rating}).$$

Here each catalog tuple specifies an item for present, its type (*e.g.,* jewelry, book), price, and the number of the item in stock. Purchase history is recorded by relation history: a history tuple indicates that a buyer bought an item for a recipient specified by gender, age and relationship with the buyer, for an event (*e.g.,* birthday, wedding, holiday), as well as rating given by the buyer in the range of [1,5].

[1] http://www.findgift.com.

Peter wants to use the engine to find a Christmas gift for his 14 year-old niece Grace, in the price range of [$20, $30]. His request can be converted to a query Q_0 defined on database D_0. The relevance $\delta_{\mathsf{rel}}(t, Q_0)$ of a tuple t returned by $Q_0(D_0)$ can be assessed by using the information from relation history, by taking into account previous presents purchased for girls of 12–16 year old by the girls' relatives for holidays, as well as the rating by those buyers. The distance (diversity) $\delta_{\mathsf{dis}}(t_1, t_2)$ between two items t_1 and t_2 returned by $Q_0(D_0)$ can be estimated by considering the differences between their types. Peter wants the system to recommend a set of 10 items from $Q_0(D_0)$ such that on one hand, those items are as fit as possible as a Christmas present for a teenage girl, and on the other hand, are as dissimilar as possible to cover a wide range of choices.

The computational complexity of processing such requests depends on both the queries expressing users' requests and the objective function used by the system.

(1) *Query languages*. Query Q_0 can be expressed as a conjunctive query (CQ). Nonetheless, if Peter wants a new gift that is different from previous gifts he gave to Grace, we need first-order logic (FO) to express Q_0, by using negation on relation history. In practice one cannot expect that $Q_0(D_0)$ is already computed when Peter submits his request. As remarked earlier, it is too costly to compute $Q_0(D_0)$ first and then pick a top set of k items from $Q_0(D_0)$. Instead, we want to embed result diversification in the evaluation of Q_0, and ideally, find a satisfactory set of k items *without* retrieving the entire set $Q_0(D_0)$ and paying its cost. A question concerns what difference CQ and FO make on the complexity of processing such requests when $Q_0(D_0)$ is not necessarily available. One naturally wants to know whether the complexity is *introduced by the query languages or is inherent to result diversification*.

(2) *Objective functions*. Consider the objective function by max-sum diversification proposed in [Gollapudi and Sharma 2009] and revised in [Vieira et al. 2011]:

$$F_{\mathsf{MS}}(U) = (k-1)(1-\lambda) \cdot \sum_{t \in U} \delta_{\mathsf{rel}}(t, Q) + \lambda \cdot \sum_{t, t' \in U} \delta_{\mathsf{dis}}(t, t'),$$

where U is a set of tuples in $Q(D)$. To assess the diversity, $F_{\mathsf{MS}}(U)$ only requires to compute $\delta_{\mathsf{dis}}(t, t')$ for t and t' in a given k-element set $U \subseteq Q(D)$. Similarly, the objective function by max-min diversification is defined as [Gollapudi and Sharma 2009]:

$$F_{\mathsf{MM}}(U) = (1-\lambda) \cdot \min_{t \in U} \delta_{\mathsf{rel}}(t) + \lambda \cdot \min_{t, t' \in U, t \neq t'} \delta_{\mathsf{dis}}(t, t').$$

In contrast, consider the mono-objective formulation of [Gollapudi and Sharma 2009]:

$$F_{\mathsf{mono}}(U) = \sum_{t \in U} \Big((1-\lambda) \cdot \delta_{\mathsf{rel}}(t, Q) + \frac{\lambda}{|Q(D)| - 1} \cdot \sum_{t' \in Q(D)} \delta_{\mathsf{dis}}(t, t') \Big).$$

It asks for $\delta_{\mathsf{dis}}(t, t')$ for each $t \in U$ and for *all* $t' \in Q(D)$, *i.e.,* the *average dissimilarity w.r.t.* all other results in $Q(D)$ [Minack et al. 2009]. The question is what *different impacts* $F_{\mathsf{MS}}(\cdot)$, $F_{\mathsf{MM}}(\cdot)$ and $F_{\mathsf{mono}}(\cdot)$ have on the complexity of diversification. □

To the best of our knowledge, no prior work has answered these questions. These issues require a full treatment for different query languages and objective functions, to find out *where the complexity of query result diversification arises*.

Contributions. We study several fundamental problems in connection with result diversification for relational queries, and establish their upper bounds and lower bounds, all matching, for a variety of query languages and objective functions.

Diversification problems. We identify three problems for query result diversification. Given a query Q, a database D, an objective function $F(\cdot)$, and a positive integer k,

(1) *the query result diversification problem* (QRD) is a decision problem to determine

whether there exists a k-element set $U \subseteq Q(D)$ such that $F(U) \geq B$ for a given bound B, *i.e.,* whether there exists a set U that satisfies the users' need at all;

(2) *the diversity ranking problem* (DRP) is to decide whether a given k-element set $U \subseteq Q(D)$ is among top-r ranked sets, such that there exist no more than $r-1$ sets $S \subseteq Q(D)$ of k elements with $F(S) > F(U)$; as advocated in [Jin and Patel 2011], a decision procedure for DRP can help us assess how well a given k-element set U satisfies the users' request, and help vendors evaluate their products *w.r.t.* users' need; and

(3) *the result diversity counting problem* (RDC) is to count the number of k-element sets $U \subseteq Q(D)$ such that $F(U) \geq B$ for a given bound B. It is a *counting problem* that helps us find out how many k-element sets can be extracted from $Q(D)$ and be suggested to the users, and provide a guidance for recommender systems to adjust their stock.

Complexity results. For all these problems we establish their *combined complexity* and *data complexity* (*i.e.,* when both data D and query Q may vary, and when Q is fixed while D may vary, respectively; see [Abiteboul et al. 1995]). We parameterize these problems with various query languages, including conjunctive queries (CQ), unions of conjunctive queries (UCQ), positive existential FO queries ($\exists$FO$^+$) and first-order logic queries (FO) [Abiteboul et al. 1995], all with built-in predicates $=, \neq, <, \leq, >, \geq$. These languages have been used in query result diversification tools, *e.g.,* CQ [Chen and Li 2007], $\exists$FO$^+$ [Vee et al. 2008] and FO [Demidova et al. 2010]. For each of these query languages, we study these problems with each of the objective functions proposed by [Gollapudi and Sharma 2009], *i.e.,* objective functions defined in terms of max-sum diversification, max-min diversification, and mono-objective formulation.

We provide a comprehensive account of upper and lower bounds for these problems, *all matching* when the problems are intractable; that is, we show that such a problem is C-hard and is in C for a complexity class C that is NP or beyond in the polynomial hierarchy. We also study special cases of these problems, such as when either only diversity or only relevance is considered, when Q is an identity query, and when k is a predefined constant. We identify practical tractable cases. It should be remarked that all the previous complexity results (NP-hardness) are established for a special case of QRD studied in this work only, namely, when Q is an identity query.

Compatibility constraints. We also re-investigate these problems in the presence of *compatibility constraints* [Koutrika et al. 2009; Lappas et al. 2009; Parameswaran et al. 2010; Parameswaran et al. 2011; Xie et al. 2012]. Such constraints are defined on *a set* U of top-k items, to specify what items have to be taken together and what items have conflict with each other, among other things. The need for such constraints is evident in practice. For instance, when Peter buys a Christmas gift for Grace, he also wants to buy a Christmas card together; when one selects a course A for an undergraduate package, she has to include all the prerequisites of A in the package [Koutrika et al. 2009; Parameswaran et al. 2010]; and when one forms a basketball team, he would like to get recommendation for a center, two forwards and two point guards [Lappas et al. 2009]. No matter how important, however, few previous work has studied the impact of compatibility constraints on the analyses of query result diversification.

We propose a class $\mathcal{C}_m$ of compatibility constraints for query result diversification, which suffices to express compatibility requirements we commonly encounter in practice. The constraints of $\mathcal{C}_m$ are of a restricted form of tuple generating dependencies (see, *e.g.,* [Abiteboul et al. 1995]), and can be validated in PTIME, *i.e.,* given a set Σ of constraints in $\mathcal{C}_m$ and a dataset U, it is in PTIME to decide whether U satisfies Σ. We investigate the impact of such constraints on the combined complexity and data complexity of QRD, DRP and RDC, for query languages ranging over CQ, UCQ, $\exists$FO$^+$and FO, and when the objective function is F_{MS}, F_{MM} or F_{mono}. We also study these problems in all the special settings mentioned above, when a set of compatibility constraints of $\mathcal{C}_m$ is additionally imposed on the selected sets of query results.

Impact. These results tell us where the complexity arises (see Tables I, II and III in Section 10 for a detailed summary of the complexity bounds).

(1) *Query languages* $\mathcal{L}_Q$. Query languages may dominate the combined complexity of result diversification. For objective functions defined in terms of max-sum or max-min diversification, QRD, DRP and RDC are NP-complete, coNP-complete and $\#\cdot$NP-complete, respectively, when $\mathcal{L}_Q$ is CQ. In contrast, when it comes to FO, these problems become PSPACE-complete, PSPACE-complete and $\#\cdot$PSPACE-complete, respectively. This said, the presence of disjunction in $\mathcal{L}_Q$ does not complicate the diversification analyses. Indeed, these problems remain NP-complete, coNP-complete and $\#\cdot$NP-complete, respectively, when $\mathcal{L}_Q$ is either UCQ or $\exists\mathrm{FO}^+$.

In contrast, different query languages have no impact on the data complexity of these problems, as expected. Indeed, for max-sum or max-min diversification, QRD, DRP and RDC are NP-complete, coNP-complete and $\#\cdot$NP-complete, respectively, and for mono-objective formulation, they are in PTIME (polynomial time), PTIME and $\#\cdot$P-complete, respectively, no matter whether $\mathcal{L}_Q$ is CQ or FO. Intuitively, a naive algorithm for QRD works in two steps: first compute $Q(D)$, and then finds whether there exists a k-element set U from $Q(D)$ such that $F(U) \geq B$; similarly for DRP and RDC. When Q is fixed as in the setting of data complexity analysis, $Q(D)$ is in PTIME regardless of what query language $\mathcal{L}_Q$ we use to express Q. The data complexity of the problems arises from the second step, *i.e.,* the diversification computation.

(2) *Objective functions* $F(\cdot)$. When $F(\cdot)$ is F_{mono}, however, the objective function *dominates* the complexity: QRD, DRP and RDC are PSPACE-complete, PSPACE-complete and $\#\cdot$PSPACE-complete, respectively, no matter whether $\mathcal{L}_Q$ is CQ or FO. Contrast these with their counterparts given above for F_{MS} and F_{MM}. The impact of $F(\cdot)$ is even more evident on the data complexity. As remarked earlier, for F_{MS} and F_{MM}, these problems are NP-complete, coNP-complete and $\#\cdot$NP-complete, respectively, for data complexity, whereas they are in PTIME, PTIME and $\#\cdot$P-complete, respectively, for F_{mono}.

(3) *Diversity vs. relevance*. The complexity is mostly introduced by the diversity requirement. This is consistent with the observation of [Vieira et al. 2011], which studied a special case of QRD when $F(\cdot)$ is F_{MS}. Indeed, when the relevance function $\delta_{\mathsf{rel}}(\cdot,\cdot)$ is absent, the combined and data complexity bounds remain unchanged for all these problems, for any of the three objective functions. In contrast, when the distance function $\delta_{\mathsf{dis}}(\cdot,\cdot)$ is dropped, QRD and DRP become tractable when data complexity is considered. Moreover, for F_{mono}, when $\delta_{\mathsf{dis}}(\cdot,\cdot)$ is absent, the combined complexity of QRD, DRP and RDC becomes NP-complete, coNP-complete and $\#\cdot$NP-complete, down from PSPACE-complete, PSPACE-complete and $\#\cdot$PSPACE-complete, respectively. In particular, for F_{MS} (resp. F_{MM}), one can draw an analogy between $\delta_{\mathsf{rel}}(\cdot,\cdot)$ and sorting with a target weight, and between $\delta_{\mathsf{dis}}(\cdot,\cdot)$ and partitioning with dispersed objects, which are requirements of the (resp. Maximum) Dispersion Problem [Prokopyev et al. 2009].

(4) *Compatibility constraints*. Although the constraints of $\mathcal{C}_m$ are simple enough to be validated in PTIME, their presence complicates the analyses of QRD, DRP and RDC, to an extent. Indeed, all tractable cases of these problems in the absence of compatibility constraints become intractable when constraints of $\mathcal{C}_m$ are present. These include (a) the data complexity analyses of QRD, DRP and RDC when $F(\cdot)$ is F_{mono}, or when $F(\cdot)$ is F_{MS} or F_{MM} defined in terms of the relevance function $\delta_{\mathsf{rel}}(\cdot,\cdot)$ only; and (b) the combined complexity of these problems for identity queries when $F(\cdot)$ is F_{mono}. The

only exception is the case when the bound k on the number of selected tuples is a constant; in this case, the data complexity analyses of these problems are tractable no matter whether the constraints of $\mathcal{C}_m$ are present or not.

These results reveal the impacts of various factors on the complexity of query result diversification. In particular, the results tell us that the complexity of these problems for CQ, UCQ and $\exists$FO$^+$may be *inherent to* result diversification itself, rather than a consequence of the complexity of the query languages. From the results we can see that these problems are intricate and mostly intractable. This highlights the need for developing efficient heuristic (approximation whenever possible) algorithms for them.

Organization. We discuss related work in Section 2, and present a general model for query result diversification in Section 3. Problems QRD, DRP and RDC are formulated in Section 4, and their combined complexity and data complexity are established in Sections 5, 6 and 7, respectively. Section 8 studies special cases of these problems, and Section 9 revisits these problems in the presence of compatibility constraints. Finally, Section 10 identifies directions for future work. Due to the space constraint we defer the proofs of the results of Sections 8 and 9 to the electronic appendix.

2. RELATED WORK

This paper is an extension of our earlier work [vld] by including the following. (1) Detailed proofs of all the results (Sections 5, 6, 7 and 8), which were not presented in [vld]. A variety of techniques are used to prove these results, including counting arguments, a wide range of reductions and constructive proofs with algorithms. (2) An extension of the query result diversification model with compatibility constraints, and the (combined and data) complexity of all these problems in the presence of compatibility constraints (Section 9). These are among the first results for incorporating compatibility constraints into query results diversification.

This work is also related to prior work on result diversification (for search and queries), recommender systems and top-k query answering, discussed as follows.

Diversification. Diversification has been studied for Web search [Agrawal et al. 2009; Borodin et al. 2012; Capannini et al. 2011; Gollapudi and Sharma 2009; Vieira et al. 2011], recommender systems [Yu et al. 2009a; 2009b; Zhang and Hurley 2008; Ziegler et al. 2005], structured databases [Demidova et al. 2010; Fraternali et al. 2012; Liu et al. 2009; Vee et al. 2008] and sponsored-search advertising [Feuerstein et al. 2007] (see [Drosou and Pitoura 2010; Minack et al. 2009] for surveys). The previous work has mostly focused on metrics for assessing relevance and diversity, and optimization techniques for computing diverse answers. The prior work often adopts specific objective functions based on the similarity of, *e.g.,* taxonomy [Ziegler et al. 2005], explanations [Yu et al. 2009a], features [Vee et al. 2008] or locations [Fraternali et al. 2012]. A general model for result diversification was proposed in [Gollapudi and Sharma 2009] based on an axiom system, along with the three objective functions mentioned earlier. A minor revision of max-sum diversification of [Gollapudi and Sharma 2009] was presented in [Vieira et al. 2011]. This work extends the model of [Gollapudi and Sharma 2009] by incorporating queries (and compatibility constraints). Like in [Borodin et al. 2012], we focus on the objective functions proposed in [Gollapudi and Sharma 2009].

The complexity of result diversification has been studied in [Agrawal et al. 2009; Gollapudi and Sharma 2009; Liu et al. 2009; Stefanidis et al. 2010; Vieira et al. 2011], which differ from this work in the following.

(1) The previous work provided lower bounds (NP-hardness) but stopped short of giving

a matching upper bound. In contrast, we provide a complete picture of matching upper and lower bounds, for both combined and data complexity.

(2) The prior work assumed that the search space $Q(D)$ is already computed, and is taken as input. As remarked earlier, this assumption is not very realistic in practice. In contrast, we treat Q and D as input instead of $Q(D)$, and investigate the impact of query languages on the complexity of diversification. As will be seen later, the complexity bounds of these problems when $Q(D)$ is not available is quite different from their counterparts when $Q(D)$ is assumed in place (*i.e.,* when Q is an identity query).

(3) The previous work focused on a special cases of QRD, when Q is an identity query. It is one of the special cases studied in Section 8 of this paper. Note that the intractability of QRD for max-sum or max-min diversification given in the prior work [Drosou and Pitoura 2009; Gollapudi and Sharma 2009; Vieira et al. 2011] may be adapted to establish the data complexity of QRD in these settings. Nonetheless, the detailed proofs are not given in those papers. Further, for mono-objective formulation, no previous work has studied the complexity of QRD for identity queries, which will be shown in PTIME in this work. Moreover, we are not aware of any complexity results for DRP and RDC published by previous work, although DRP was advocated in [Jin and Patel 2011].

(4) This work also considers several special cases of diversification (Section 8), to identify tractable cases and the impact of diversity and relevance requirements on the complexity of the diversification analyses. Moreover, we also study diversification in the presence of compatibility constraints, about which we are not aware of any prior work.

Recommender problems. Recommender systems (*a.k.a.* recommender engines and recommendation platforms) are to recommend information items or social elements that are likely to be of interest to users (see [Adomavicius and Tuzhilin 2005] for a survey). There has been a host of work on recommender systems [Deng et al. 2012; Amer-Yahia 2011; Lappas et al. 2009; Koutrika et al. 2009; Parameswaran et al. 2011; Xie et al. 2012], studying item and package recommendation. Given a query Q, a database D of items and a utility (scoring) function $f(\cdot)$ defined on items, *item recommendation* is to find top-k items from $Q(D)$ ranked by $f(\cdot)$, for a given positive integer k. *Package recommendation* takes as additional input a set Σ of compatibility constraints, two functions $\mathsf{cost}(\cdot)$ and $\mathsf{val}(\cdot)$ defined on sets of items, and a bound C. It is to find top-k packages of items such that each package satisfies Σ, its cost does not exceed C, and its val is among the k highest. Here a package is *a set of items* that has a *variable* size.

There is an intimate connection between recommendation and diversification: both aim to recommend top-k (sets of) items from the result $Q(D)$ of query Q in D. Moreover, diversification has been used in recommender systems to rectify the problem of retrieving too homogeneous results. However, there are subtle differences between them.

(1) Item recommendation is a single-criterion optimization problem based on a utility function $f(\cdot)$ defined on individual items. In contrast, query result diversification is a bi-criteria optimization problem based on a relevance function $\delta_{\mathsf{rel}}(\cdot,\cdot)$ and a distance function $\delta_{\mathsf{dis}}(\cdot,\cdot)$ defined on *sets of items*. In particular, the distance function $\delta_{\mathsf{dis}}(U)$ assesses the diversity of elements in a set U, and is not expressible as a utility function.

(2) Package recommendation is to find top-k sets of items with *variable sizes*, which are ranked by $\mathsf{val}(\cdot)$, subject to compatibility constraints Σ and aggregate constraints defined in terms of $\mathsf{cost}(\cdot)$ and bound C, where $\mathsf{cost}(\cdot)$ and $\mathsf{val}(\cdot)$ are generic PTIME computable functions [Deng et al. 2012]. In contrast, query result diversification is to find *a single set of k items*, based on a *particular* objective function $F(\cdot)$. When $F(\cdot)$ is max-sum or max-min diversification, diversification can be viewed as a special case of

package recommendation for finding a single set of a fixed size k, based on a particular $F(\cdot)$, and in the absence of aggregate constraints. As a consequence of the specific restrictions of $F(\cdot)$, the lower bounds developed for package recommendation *do not* carry over to its counterpart for diversification, and conversely, the upper bounds for diversification *may not be* tight for package recommendation. When $F(\cdot)$ is mono-objective, $F(U)$ is *not* even expressible in the model of recommendation, since it assesses the diversity of elements in a set U with *all* tuples in $Q(D)$, and is not in PTIME in $|U|$.

There has been work on the complexity of recommendation analyses [Amer-Yahia et al. 2013; Deng et al. 2012; Lappas et al. 2009; Koutrika et al. 2009; Parameswaran et al. 2011; Xie et al. 2012]. In addition to different settings of recommendation and diversification remarked earlier, this work differs from the prior work in the following.

(3) Problems QRD and DRP studied in this paper have not been considered in the previous work for recommendation. This said, the results of this work on these problems may be of interest to the study of recommendation.

(4) Problem RDC considered here is similar to a counting problem studied in [Deng et al. 2012] for recommendation. However, given the different settings remarked earlier, RDC differs from that counting problem from complexity bounds to proofs. Indeed, the counting problem for recommendation is $\#\cdot$coNP-complete when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$ [Deng et al. 2012]. In contrast, as will be seen in Section 7, for the same query languages, (a) RDC is $\#\cdot$NP-complete when $F(\cdot)$ is F_{MS} or F_{MM}, while $\#\cdot$coNP = $\#\cdot$NP if and only if P = NP [Durand et al. 2005]; and (b) RDC is $\#\cdot$PSPACE-complete when $F(\cdot)$ is F_{mono}, substantially more intriguing than the problem studied in [Deng et al. 2012]. Furthermore, the proofs of this paper have to be tailored to the three objective functions, as opposed to the proofs of [Deng et al. 2012]. Indeed, the proofs for F_{MS} and F_{MM} are quite different from their counterparts for F_{mono}, as indicated by the different combined complexity bounds in these settings.

Compatibility constraints have been studied for package recommendation [Amer-Yahia et al. 2013; Koutrika et al. 2009; Lappas et al. 2009; Parameswaran et al. 2010; Parameswaran et al. 2011; Xie et al. 2012]. As remarked above, the prior results do not carry over to the diversification analysis in the presence of compatibility constraints.

Top-k query answering. Top-k query answering is to retrieve top-k tuples from query results, ranked by a scoring function. It typically assumes that the attributes of tuples are already sorted, and studies how to combine different ratings of the attributes for the same tuple based on a (monotonic) scoring function. A number of top-k query evaluation algorithms have been developed (*e.g.,* [Fagin et al. 2003; Jin and Patel 2011; Li et al. 2005; Schnaitter and Polyzotis 2008]; see [Ilyas et al. 2008] for a survey), focusing on how to achieve early termination and reduce random access. This work differs from the prior work in the following. (a) A scoring function for top-k query answering is defined on individual items, as opposed to the distance function $\delta_{\mathsf{dis}}(\cdot)$ and the objective function $F(\cdot)$ defined on sets of items. (b) We focus on the complexity of diversification problems rather than the efficiency or optimization of query evaluation.

3. DIVERSIFICATION AND OBJECTIVE FUNCTIONS

We first present a model for query result diversification, by extending the model of [Gollapudi and Sharma 2009]. We then review the three objective functions proposed by [Gollapudi and Sharma 2009], which are used to define diversification.

3.1. Query Result Diversification

As remarked earlier, query result diversification aims to improve user satisfaction when computing answers to a query Q in a database D. We specify database D with a relational schema $\mathcal{R} = (R_1, \ldots, R_n)$, where each relation schema R_i is defined over a fixed set of attributes. We consider query Q expressed in a query language $\mathcal{L}_Q$.

Diversification. Given Q, D, a positive integer k and an objective function $F(\cdot)$, *query result diversification* aims to find a set $U \subseteq Q(D)$ such that (a) $|U| = k$, and (b) $F(U)$ is maximum, *i.e.,* for all other sets $U' \subseteq Q(D)$, if $|U'| = k$ then $F(U) \geq F(U')$. Here $F(\cdot)$ is an objective function defined on *sets* of tuples of R_Q, where R_Q denotes the schema of query result $Q(D)$, such that given any set U of tuples of R_Q, $F(U)$ returns a non-negative real number. In other words, $F(\cdot)$ is defined on subsets $U \subseteq Q(D)$. We write $F(\cdot)$ as F when it is clear from the context.

Intuitively, query result diversification is to retrieve a set U of k answers to Q in D such that the tuples in U are as relevant as possible to Q and meanwhile, as diverse as possible. It extends the notion of result diversification given in [Gollapudi and Sharma 2009] by taking query Q and D as input, rather than assuming that $Q(D)$ is already computed. The notion of [Gollapudi and Sharma 2009] is a special case of query result diversification, when Q is an identity query, *i.e.,* when $Q(D) = D$ is given as input.

Query result diversification is a bi-criteria optimization problem characterized by objective function F which is defined in terms of a relevance function $\delta_{\mathsf{rel}}(\cdot,\cdot)$ and a distance function $\delta_{\mathsf{dis}}(\cdot,\cdot)$; these functions are presented as follows.

Relevance functions and distance functions. A *relevance function* $\delta_{\mathsf{rel}}(\cdot,\cdot)$ is defined on tuples of schema R_Q and queries in $\mathcal{L}_Q$. It specifies the relevance of a tuple t of R_Q to a query $Q \in \mathcal{L}_Q$. More specifically, $\delta_{\mathsf{rel}}(t, Q)$ is a non-negative real number such that the larger $\delta_{\mathsf{rel}}(t, Q)$ is, the more relevant the answer t is to query Q.

A *distance function* $\delta_{\mathsf{dis}}(\cdot,\cdot)$ is a binary function defined on tuples of schema R_Q. It specifies the diversity between two tuples $t, s \in Q(D)$: $\delta_{\mathsf{dis}}(t, s)$ is a non-negative real number such that the larger $\delta_{\mathsf{dis}}(t, s)$ is, the more diverse (dissimilar) t and s are to each other. We assume that $\delta_{\mathsf{dis}}(\cdot,\cdot)$ is symmetric, *i.e.,* $\delta_{\mathsf{dis}}(t, s) = \delta_{\mathsf{dis}}(s, t)$ for all tuples t, s of R_Q. Moreover, $\delta_{\mathsf{dis}}(t, t) = 0$, *i.e.,* the distance between a tuple and itself is 0.

We simply assume that $\delta_{\mathsf{rel}}(\cdot,\cdot)$ and $\delta_{\mathsf{dis}}(\cdot,\cdot)$ are PTIME computable functions, as commonly found in practice, and focus on their generic properties. We also write $\delta_{\mathsf{rel}}(\cdot,\cdot)$ and $\delta_{\mathsf{dis}}(\cdot,\cdot)$ as δ_{rel} and δ_{dis}, respectively, if it is clear from the context.

Example 3.1. Recall the request of Peter for shopping a gift for Grace described in Example 1.1. The request can be expressed as a query Q_0 in FO as follows:

$$\begin{aligned} Q_0(n) = \exists\, t, p, s\ \big(&\mathsf{catalog}(n, t, p, s) \wedge p \leq 30 \wedge p \geq 20 \wedge \\ &\forall n', b, r, g, a, x, e, y\ \neg(\mathsf{history}(n', b, r, g, a, x, e, y) \wedge \\ &\qquad b = \mathrm{id}_P \wedge r = \text{“Grace”} \wedge n = n')\big), \end{aligned}$$

where id_P denotes Peter's buyer id. The query selects such gifts in the price range $[\$20, \$30]$ that have not been purchased by Peter for Grace earlier.

As remarked in Example 1.1, for each gift $t \in Q_0(D_0)$, the relevance $\delta_{\mathsf{rel}}(t, Q_0)$ of t to Q_0 can be assessed in terms of the rating of t if t appears in the history relation. For instance, $\delta_{\mathsf{rel}}(t, Q_0)$ is high if t was presented as a gift for a girl of age [11, 14] by a relative for a holiday, and was rated high. If t is not in history, $\delta_{\mathsf{rel}}(t, Q_0)$ takes a default value.

For tuples $t, s \in Q_0(D_0)$, $\delta_{\mathsf{dis}}(t, s)$ can be defined in terms of the difference between their types, *e.g.,* $\delta_{\mathsf{dis}}(t, s) = 2$ if t is in the “artsy” category and s is “educational”, and $\delta_{\mathsf{dis}}(t, s) = 1$ if t is of type “jewelry” and s is of type “fashion”. The types can be classified into various categories and brands, and $\delta_{\mathsf{dis}}(t, s)$ is defined accordingly. □

3.2. Objective Functions

An objective function F is defined by means of relevance function δ_{rel} and distance function δ_{dis}. Like in [Borodin et al. 2012], we focus on the objective functions proposed by [Gollapudi and Sharma 2009] in this work.

Consider δ_{rel} and δ_{dis}, a parameter $\lambda \in [0, 1]$ to balance relevance and diversity, a query Q, a database D and a positive integer k. Let $U \subseteq Q(D)$ be a set of tuples with $|U| = k$. A minor revision of max-sum diversification of [Gollapudi and Sharma 2009] was given in [Vieira et al. 2011] by associating $(1 - \lambda)$ with the relevance component, which allows us to study two extreme cases: diversity only (*i.e.,* when $\lambda = 1$), and relevance only (*i.e.,* when $\lambda = 0$). Along the same line as [Vieira et al. 2011], we consider minor variations of the max-min diversification and mono-objective functions of [Gollapudi and Sharma 2009]. We present the revised objective functions as follows.

Max-sum diversification. The first objective is to maximize the sum of the relevance and dissimilarity of the selected set U [Gollapudi and Sharma 2009; Vieira et al. 2011]:

$$F_{\mathsf{MS}}(U) = (k-1)(1-\lambda) \cdot \sum_{t \in U} \delta_{\mathsf{rel}}(t, Q) + \lambda \cdot \sum_{t, t' \in U} \delta_{\mathsf{dis}}(t, t').$$

Here $F_{\mathsf{MS}}(U)$ measures both the relevance of the tuples in U to query Q, and the diversity among the k tuples in U. Following [Gollapudi and Sharma 2009], we scale up the two components δ_{rel} and δ_{dis} by using $k - 1$ since the relevance sum ranges over k numbers while the diversity sum is over $k(k-1)$ numbers (note that the same effect may also be achieved by tailoring λ; we adopt $k - 1$ here to simplify the discussion).

As observed by [Gollapudi and Sharma 2009; Vieira et al. 2011], when the objective function is F_{MS}, result diversification can be modeled as the Dispersion Problem studied in operations research [Prokopyev et al. 2009], when Q is an identity query.

Max-min diversification. The second objective is to maximize the minimum relevance and dissimilarity of the selected set [Gollapudi and Sharma 2009]:

$$F_{\mathsf{MM}}(U) = (1-\lambda) \cdot \min_{t \in U} \delta_{\mathsf{rel}}(t, Q) + \lambda \cdot \min_{t, t' \in U, t \neq t'} \delta_{\mathsf{dis}}(t, t').$$

Here $F_{\mathsf{MM}}(U)$ is computed in terms of both the minimum relevance of the k tuple in U to query Q, and the minimum distance between any two tuples in U. In contrast to $F_{\mathsf{MS}}(U)$, $F_{\mathsf{MM}}(U)$ tends to penalize U that includes a single item irrelevant to Q, or that contains a pair of homogeneous items, although all the rest are diverse. As shown in [Gollapudi and Sharma 2009], diversification by F_{MM} can be expressed as the Maxmin Dispersion Problem [Prokopyev et al. 2009] if Q is an identity query.

Mono-objective formulation. The last one is given as [Gollapudi and Sharma 2009]:

$$F_{\mathsf{mono}}(U) = \sum_{t \in U} \big((1-\lambda) \cdot \delta_{\mathsf{rel}}(t, Q) + \frac{\lambda}{|Q(D)| - 1} \cdot \sum_{t' \in Q(D)} \delta_{\mathsf{dis}}(t, t')\big).$$

As opposed to $F_{\mathsf{MS}}(U)$ and $F_{\mathsf{MM}}(U)$ that compute intro-list diversity, $F_{\mathsf{mono}}(U)$ measures the "global" diversity of each tuple $t \in U$ by taking the mean of its distance to *all* tuples in the entire set $Q(D)$, rather than its distances to the tuples in the selected set U [Gollapudi and Sharma 2009]. It computes the average dissimilarity of tuples in U by comparing them *uniformly* with all other results in $Q(D)$ [Minack et al. 2009], to assess *the novelty and coverage* of the items in U. In this formulation, the distance function behaves similarly to the relevance function; hence it is named "mono". In contrast with F_{MS} and F_{MM}, $F_{\mathsf{mono}}(U)$ does not reduce to facility dispersion.

Example 3.2. Consider the query Q_0, database D_0, and the relevance and distance functions δ_{rel} and δ_{dis} described in Example 3.1. Assume that $k = 10$. Then

(1) with F_{MS}, query result diversification aims to find a set U_1 of 10 gifts from the query

result $Q_0(D_0)$ such that the weighted sum of the relevance values of the selected gifts in U_1 to Q_0 and the dissimilarity values among the gifts in U_1 is maximum.

(2) The objective function F_{MM} is to find a set U_2 of 10 gifts from $Q_0(D_0)$ such that the weighted sum of the minimum relevance of the gifts in U_2 to Q_0 and the minimum distance between pairs of gifts in U_2 is maximum.

(3) The objective function F_{mono} is to find a set U_3 of 10 gifts from $Q_0(D_0)$ such that the weighted sum of the relevance values of the gifts in U_3 to Q_0 and the mean of the distances between the selected gifts in U_3 and *all candidate* gifts in the entire set $Q_0(D_0)$ is maximized. In particular, here the diversity criterion is to assess the coverage of various gifts in the entire set $Q_0(D_0)$ by the set U chosen. □

Remarks. Observe the following.

(1) Objective functions F_{MS}, F_{MM} and F_{mono} are defined in terms of two criteria: relevance δ_{rel} and diversity δ_{dis}. The larger the parameter λ is, the more weight we place on the diversity of the results selected. When $\lambda = 0$, F_{MS}, F_{MM} and F_{mono} measure the relevance only. On the other hand, when $\lambda = 1$, these objective functions are defined in terms of δ_{dis} only and assess the diversity alone.

(2) For a given set $U \subseteq Q(D)$, $F_{\mathsf{MS}}(U)$ and $F_{\mathsf{MM}}(U)$ are PTIME computable as long as δ_{rel} and δ_{dis} are PTIME computable. In contrast, when it comes to mono-objective, $F_{\mathsf{mono}}(U)$ may *not* be PTIME computable when Q and D are given as input but $Q(D)$ is not assumed available, as commonly found in practice. Indeed, for each tuple $t \in U$, $F_{\mathsf{mono}}(U)$ has to compute $\delta_{\mathsf{dis}}(t, t')$ when t' ranges over *all* tuples in $Q(D)$.

4. REASONING ABOUT RESULT DIVERSIFICATION

In this section we first identify three problems in connection with query result diversification, for which the complexity will be provided in the next five sections. We then demonstrate possible applications of the complexity analyses of these problems.

4.1. Decision and Counting Problems

Consider a database D, a query Q in a language $\mathcal{L}_Q$, a positive integer k, and an objective function F defined with relevance and distance functions δ_{rel} and δ_{dis}.

The query result diversification problem. We start with a decision problem, referred to as *the query result diversification problem* and denoted by $\mathsf{QRD}(\mathcal{L}_Q, F)$. To formulate this problem, we need the following notations.

We call a set $U \subseteq Q(D)$ a *candidate set for* (Q, D, k) if $|U| = k$. Given a real number B as a bound, we refer to a candidate set U as *a valid set for* (Q, D, k, F, B) if $F(U) \geq B$. That is, the F value of U is large enough to meet the objective B.

Given these, $\mathsf{QRD}(\mathcal{L}_Q, F)$ is stated as follows.

$\mathsf{QRD}(\mathcal{L}_Q, F)$:	The *query result diversification problem*.
INPUT:	A database D, a query $Q \in \mathcal{L}_Q$, an objective function F, a real number B and a positive integer $k \geq 1$.
QUESTION:	Does there exist a valid set for (Q, D, k, F, B)?

Observe that $\mathsf{QRD}(\mathcal{L}_Q, F)$ is the decision version of the function problem for computing a top-ranked set U based on F, and is fundamental to understanding the complexity of query result diversification. As remarked earlier, we simply consider generic PTIME functions δ_{rel} and δ_{dis} when defining F.

The diversity ranking problem. In practice, given a candidate set U picked by users or produced by a system, we want to assess how well U meets a diversification objective and hence, satisfies the users' need. This suggests that we study another decision problem, referred to as *the diversity ranking problem* and denoted by $\mathsf{DRP}(\mathcal{L}_Q, F)$, to assess the rank of a given candidate set based on F.

To state this problem, we use the following notion of ranks. Consider a candidate set U and a positive integer r. We say that the *rank* of U is r, denoted by $\mathsf{rank}(U) = r$, if there exists a collection $\mathcal{S}$ of $r-1$ distinct candidate sets for (Q, D, k) such that (a) for all $S \in \mathcal{S}$, $F(S) > F(U)$; and (b) for any candidate set S' for (Q, D, k), if $S' \notin \mathcal{S}$, then $F(U) \geq F(S')$. That is, there exist exactly $r-1$ candidates sets for (Q, D, k) that are ranked above U based on F. Obviously, the less $\mathsf{rank}(U)$ is, the higher U is ranked.

Assume a positive integer r that is a constant. We state $\mathsf{DRP}(\mathcal{L}_Q, F)$ as follows.

$\mathsf{DRP}(\mathcal{L}_Q, F)$:	The *diversity ranking problem*.
INPUT:	A database D, a query $Q \in \mathcal{L}_Q$, an objective function F, a positive integer $k \geq 1$ and a candidate set U for (Q, D, k).
QUESTION:	Does $\mathsf{rank}(U) \leq r$?

This problem was advocated in [Jin and Patel 2011], but its complexity was not settled before. The need for studying this is evident, as will be elaborated in Section 4.2.

In practice, rank r is below a threshold (*e.g.,* top 20) and hence, is typically treated as a constant. One may also want to treat r as part of input rather than a constant. We will elaborate its impact on the complexity bounds of DRP in Section 6.

The result diversity counting problem. Given an objective B, one often wants to know how many valid sets are out there and hence, can be selected and recommended. This suggests that we study the counting problem below, referred to as *the result diversity counting problem* and denoted by $\mathsf{RDC}(\mathcal{L}_Q, F)$.

$\mathsf{RDC}(\mathcal{L}_Q, F)$:	*The result diversity counting problem*.
INPUT:	A database D, a query $Q \in \mathcal{L}_Q$, an objective function F, a real number B and a positive integer $k \geq 1$.
QUESTION:	How many valid sets are there for (Q, D, k, F, B)?

That is, $\mathsf{RDC}(\mathcal{L}_Q, F)$ is to count the number of candidate sets in D that satisfy the users' request. An effective counting procedure is useful in practice (see Section 4.2).

Parameters of the problems. We study these problems for (a) objective functions F ranging over max-sum diversification F_{MS}, max-min diversification F_{MM}, and mono-objective formulation F_{mono} (Section 3), and for (b) query languages $\mathcal{L}_Q$ ranging over the following (see, *e.g.,* [Abiteboul et al. 1995] for details of these languages):

(1) conjunctive queries (CQ), built up from atomic formulas with constants and variables, *i.e.,* relation atoms in database schema $\mathcal{R}$ and built-in predicates $(=, \neq, <, \leq, >, \geq)$, by closing under conjunction $\wedge$ and existential quantification $\exists$;
(2) union of conjunctive queries (UCQ) $Q_1 \cup \cdots \cup Q_r$, where Q_i is in CQ for $i \in [1, r]$;
(3) positive existential FO queries ($\exists$FO$^+$), built from atomic formulas by closing under $\wedge$, *disjunction* $\vee$ and $\exists$; and
(4) first-order logic queries (FO) built from atomic formulas using $\wedge$, $\vee$, *negation* $\neg$, $\exists$ and *universal quantification* $\forall$.

That is, FO is relational algebra, CQ is the class of SPC queries supporting selection, projection and Cartesian product, UCQ is the class of SPCU queries, and $\exists$FO$^+$is the fragment of relational algebra with selection, projection, Cartesian product and union.

To the best of our knowledge, no prior work has studied the complexity of DRP and RDC. When it comes to QRD, only a special case was studied, when $\mathcal{L}_Q$ consists of identity queries and F is F_{MS} or F_{MM}. No prior work has considered QRD when $\mathcal{L}_Q$ is CQ or beyond, or when F is F_{mono}.

4.2. Applications of Diversification Analyses

Before we establish the complexity of these problems, we first demonstrate how their analyses can be used in practice. The complexity bounds of these problems are not only of theoretical interest, but may also help practitioners when developing diversification models and algorithms. As remarked in Section 2, query result diversification is needed for Web search, recommendation systems and advertising, among other things.

Guidance for what features should be supported in a system. When developing a system that supports query result diversification, one has to decide the following. What query language should be supported? What diversification function should be adopted? Would a relevance function alone suffice in the application so that one does not have to pay the cost introduced by distance functions? Would a fixed set of queries suffice for users to express their requests? Are compatibility constraints a must? The complexity study of diversification problems in different settings may help the system vendor *strike a balance between the cost of diversification analyses and the expressive power needed*. For instance, if users of the system are to use only a fixed set of FO queries, adopt a mono-objective function and need no compatibility constraints, then the diversification analyses are in PTIME (see data complexity in Table I, Section 10). In contrast, if the system allows users to issue arbitrary CQ queries, then one should be prepared for higher cost (PSPACE-complete, Table I). In the latter case, the developers of the system should go for heuristic algorithms for diversification analyses rather than for exact PTIME algorithms, as suggested by the lower bounds of the problems.

Assessing the stock of an e-commerce system. Suppose that a recommendation system maintains a collection D of items for sale. When a decision procedure for QRD constantly fails to find desired item sets upon frequent users' requests, or a procedure for DRP finds that those items recommended by the system are often ranked low, then the manager of the system should consider adjusting the stock in D. As another example, when a procedure for RDC finds a large stock of popular items, the manager may consider advertising those items to particular user groups.

Assessing the choices of users. A user of an e-commerce system may employ a procedure for DRP to asses how good a set of items of her choice is, before she commits to the purchase. Moreover, she could use a procedure for RDC to find out different variety of choices that meet her need, and hence, choose one from them.

5. THE QUERY RESULT DIVERSIFICATION PROBLEM

We start with the decision problem QRD. We first establish its combined complexity in Section 5.1 and data complexity in Section 5.2. We will then identify and study its special cases in Section 8. The complexity bounds of QRD in various settings are depicted in Fig. 1, annotated with their corresponding theorems, where each arrow indicates how the complexity of QRD is reduced in different settings (similarly for Figures 3 and 4 to be given in Sections 6 and 7, respectively). Both *combined* complexity and *data* complexity are summarized, when $\mathcal{L}_Q$ ranges over the query languages given in Section 4 (here CQ/FO denotes either CQ or FO), objective function F is F_{MS}, F_{MM} (shown in Fig. 1(a)) or F_{mono} (Fig. 1(b)), and when certain conditions are imposed in special settings (here $\lambda = 0$ means that F is defined with a relevance function only, *constant* k indicates the setting in which the number of items to be

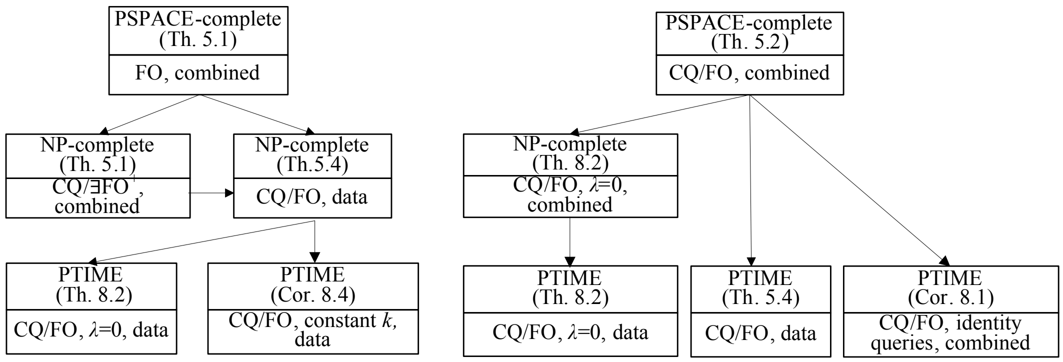


Fig. 1. The complexity bounds of QRD

selected is a constant, and *identity queries* mean that $\mathcal{L}_Q$ consists of identity queries only). They reveal the impacts of various factors on the complexity of QRD.

5.1. The Combined Complexity of QRD

Consider a naive algorithm for QRD: first compute $Q(D)$, and then rank k-element sets U of $Q(D)$ based on $F(U)$; after these steps we simply pick the top-ranked set U and check whether $F(U) \geq B$. One might think that the complexity of QRD would equal the higher complexity of the two steps. The result below tells us that this is the case when F is F_{MS} or F_{MM}. However, when F is F_{mono}, the story is quite different.

(1) When the objective is max-sum or max-min diversification, query language $\mathcal{L}_Q$ dominates the combined complexity of QRD: it is NP-complete for CQ, UCQ and $\exists$FO$^+$, but is PSPACE-complete for FO. That is, while the presence of disjunction in UCQ and $\exists$FO$^+$does not make it harder than CQ, negation in FO complicates the analysis.

(2) When F is F_{mono}, the problem becomes more intricate for CQ, UCQ and $\exists$FO$^+$: it is already PSPACE-complete, the same as its complexity for FO. Note that the membership problem for CQ is NP-complete (see the statement of the problem shortly). Moreover, F_{mono} is in PTIME after $Q(D)$ is computed (see Corollary 8.1 for details). In contrast, QRD(CQ, F_{mono}) is PSPACE-complete. Hence in this case, the complexity is inherent to query result diversification, and is not equal to the higher complexity of the two steps given above. This is because mono-objective formulation requires to aggregate distances between elements in U and all tuples in $Q(D)$, and is more costly to compute than F_{MS} and F_{MM}, as remarked in Section 3.

Below we first study QRD$(\mathcal{L}_Q, F)$ when F is F_{MS} or F_{MM}.

THEOREM 5.1. *The combined complexity of* QRD$(\mathcal{L}_Q, F_{\mathsf{MS}})$ *and* QRD$(\mathcal{L}_Q, F_{\mathsf{MM}})$ *is*

— NP-*complete when* $\mathcal{L}_Q$ *is CQ, UCQ or* $\exists$*FO*$^+$*; and*
— PSPACE-*complete when* $\mathcal{L}_Q$ *is FO.* □

To verify the lower bounds, we use reductions from the following problems.

(1) 3SAT: Given a formula $\varphi = C_1 \wedge \ldots \wedge C_l$ in which each clause C_i is a disjunction of three variables or their negations taken from $X = \{x_1, \ldots, x_m\}$, it is to decide whether φ is satisfiable. It is known that 3SAT is NP-complete (cf. [Papadimitriou 1994]).

(2) The membership problem for FO: Given an FO query Q, a database D and a tuple s, it is to decide whether $s \in Q(D)$. This problem is PSPACE-complete [Vardi 1982].

Given these, we prove Theorem 5.1 as follows.

PROOF. We study $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{MS}})$ and $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{MM}})$ first for CQ, UCQ and $\exists$FO$^+$, and then for FO.

(1) When $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$. It suffices to prove that $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MS}})$ and $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MM}})$ are NP-hard and that $\mathsf{QRD}(\exists\mathrm{FO}^+, F_{\mathsf{MS}})$ and $\mathsf{QRD}(\exists\mathrm{FO}^+, F_{\mathsf{MM}})$ are in NP.

(1.1) Lower bound. We show that $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MS}})$ and $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MM}})$ are NP-hard by reductions from the 3SAT problem, even when $\lambda = 1$ and Q is an identity query.

We first consider $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MS}})$. Given an instance $\varphi = C_1 \wedge \ldots \wedge C_l$ of 3SAT over variables $X = \{x_1, \ldots, x_m\}$, we define a database D, a fixed CQ query Q, functions δ_{rel}, δ_{dis} and F_{MS}, and constants B and k. We show that φ is satisfiable if and only if there exists a valid set U for $(Q, D, k, F_{\mathsf{MS}}, B)$. Assume *w.l.o.g.* that $l > 1$.

(1) The database D has a single relation I_C specified by $R_C(\mathsf{cid}, L_1, V_1, L_2, V_2, L_3, V_3)$ and populated as follows. For each $i \in [1, l]$, let clause C_i be $l_1^i \vee l_2^i \vee l_3^i$. For any truth assignment μ_i of variables in the literals in C_i that makes C_i true, we add a tuple $(i, x_j, v_j, x_k, v_k, x_l, v_l)$ to I_C, where $x_j = l_1^i$ if $l_1^i \in X$ and $x_j = \overline{l}_1^j$ otherwise; and $v_j = \mu_i(x_j)$; similarly for x_k, x_l and v_k and v_l. Here I_C encode all satisfying truth assignments for clauses separately. This does not give rise to an exponential blow up as each clause has only three variables. At most 8 tuples are included in I_C for each C_i.

(2) We define the query Q as the identity query on R_C instances.

(3) We define δ_{rel} to be a constant function that returns 1 for each tuple t of R_Q. For each pair of distinct tuples t and s of R_Q, we define $\delta_{\mathsf{dis}}(t, s) = 1$ in case that (a) $t[\mathsf{cid}] \neq s[\mathsf{cid}]$ (*i.e.,* t and s correspond to distinct clauses of φ) and (b) t and s have the same value for each variable appearing in both t and s. For any other pair of tuples t' and s' of R_Q, we define $\delta_{\mathsf{dis}}(t', s') = 0$. Furthermore, we use $\lambda = 1$. Then for each set U of tuples of R_Q, we have that $F_{\mathsf{MS}}(U) = \sum_{t,s \in U} \delta_{\mathsf{dis}}(t, s)$ for each set U.

(4) We use $k = l$, *i.e.,* we consider only valid sets of l tuples, one for each clause of φ. Finally, let $B = l \cdot (l - 1)$. Obviously, for any subset U of $Q(D)$ with $|U| = l$, $F_{\mathsf{MS}}(U) \geq B$ if and only if every clause has at least one satisfying assignment which is encoded by one tuple in U, and all these assignments are consistent over the variables.

We verify that φ is satisfiable if and only if there is a valid set U for $(Q, D, k, F_{\mathsf{MS}}, B)$.

$\boxed{\Rightarrow}$ Assume that φ is satisfiable. Then there exists a truth assignment μ_X^0 of X variables such that every clause C_j of φ is true by μ_X^0. Let U consist of l tuples of R_Q, one for each clause, in which the values for the variables in X agree with μ_X^0. Then $F_{\mathsf{MS}}(U) = l \cdot (l - 1) \geq B$ by the definition of F_{MS}.

$\boxed{\Leftarrow}$ Conversely, assume that φ is not satisfiable. Suppose by contradiction that there exists a set $U \subseteq Q(D)$ such that $|U| = l$ and $F_{\mathsf{MS}}(U) \geq B$. Then U consists of l tuples that corresponds to l pairwise distinct clauses in φ and all agree with values of variables in X, by the definition of δ_{dis}. Let μ_X be the truth assignment of variables in X such that for each $x_i \in X$, $\mu_X(x_i)$ equals the value of x_i in tuples in U. It is easy to see that μ_X satisfies all clauses in φ. This leads to a contradiction.

We next show that $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MM}})$ is NP-hard, also by reduction from 3SAT. Given an instance $\varphi = C_1 \wedge \ldots \wedge C_l$ of 3SAT, we construct the same D, Q, δ_{rel}, δ_{dis} as given above, and set $k = l$. Furthermore, we let $\lambda = 1$, and hence $F_{\mathsf{MM}}(U) = \min_{t,s \in U, t \neq s} \delta_{\mathsf{dis}}(t, s)$, for each set U of tuples of R_Q. Finally, we let $B = 1$. It is easy to see that μ_X^0 is a truth assignment of X variables that makes φ true if and only if there exists a set U consisting of l tuples, one for each clause, in which the values for the variables agree with μ_X^0, such that $U \subseteq Q(D)$, $|U| = k$ and $F_{\mathsf{MM}}(U) \geq B$.

(1.2) Upper bound. We show that $\mathsf{QRD}(\exists\text{FO}^+, F)$ is in NP, when F is F_{MS} or F_{MM}, by giving an NP algorithm. Given a query Q in $\exists\text{FO}^+$, a database D, a positive integer k, objective function F and a constant B, the algorithm checks whether there exists a set U valid for (Q, D, k, F B), as follows:

1. guess k CQ queries from Q, and for each CQ query, guess a tableau from D (see, *e.g.,* [Abiteboul et al. 1995] for tableaux); these tableaux yield a set $U \subseteq Q(D)$;
2. check whether $|U| = k$ and whether $F(U) \geq B$; if so, return "yes", and otherwise reject the guess and go back to step 1.

Clearly, step 2 is in PTIME since $F_{\mathsf{MS}}(U)$ and $F_{\mathsf{MM}}(U)$ and PTIME computable. Thus the algorithm is in NP, and so are $\mathsf{QRD}(\exists\text{FO}^+, F_{\mathsf{MS}})$ and $\mathsf{QRD}(\exists\text{FO}^+, F_{\mathsf{MM}})$.

(2) When $\mathcal{L}_Q$ is FO. We next study $\mathsf{QRD}(\text{FO}, F_{\mathsf{MS}})$ and $\mathsf{QRD}(\text{FO}, F_{\mathsf{MM}})$.

(2.1) Lower bound. We first show that $\mathsf{QRD}(\text{FO}, F_{\mathsf{MS}})$ is PSPACE-hard even when $\lambda = 0$ and k is a constant, by reduction from the membership problem for FO. Given an instance (Q, D, s) of the membership problem for FO, we define a database $D' = (D, I_{01})$, where $I_{01} = \{(1), (0)\}$ is a unary relation specified by schema $R_{01}(X)$, encoding the Boolean domain. Moreover, we define a query Q' in FO as follows:

$$Q'(\vec{x}, c) = Q(\vec{x}) \wedge R_{01}(c).$$

Clearly, if $s \in Q(D)$, Q' must return two tuples $(s, 1)$ and $(s, 0)$. We define $\delta_{\mathsf{rel}}((s, 1), Q')$ $= 1$, and for any other tuple t of $R_{Q'}$, $\delta_{\mathsf{rel}}(t, Q') = 0$. Furthermore, we define δ_{dis} as a constant function that returns 0 for each pair of tuples of R_Q. We use $\lambda = 0$. Then $F_{\mathsf{MS}}(U) = (k-1) \cdot \sum_{t \in U} \delta_{\mathsf{rel}}(t, Q')$ for a set U of k tuples. Finally, we let $k = 2$ and $B = 1$.

We show that $s \in Q(D)$ if and only if there exists a valid set for (Q', D', k, F_{MS}, B). First assume that $s \in Q(D)$. Then there is a valid set $U = \{(s, 1), (s, 0)\}$ for (Q', D', k, F_{MS}, B). Conversely, if $s \notin Q(D)$, then by the definition of Q', $(s, 1)$ is not in $Q'(D)$. Thus, by the definition of F_{MS}, there is no set $U \subseteq Q'(D')$ such that $|U| = 2$ and $F(U) \geq B$.

We next consider $\mathsf{QRD}(\text{FO}, F_{\mathsf{MM}})$. Given an instance (Q, D, s) of the membership problem for FO, we define the same D', Q', δ_{rel} and δ_{dis} as given above. Furthermore, we set $\lambda = 0$, $B = 1$ and $k = 1$. Then $F_{\mathsf{MM}}(U) = \min_{t \in U} \delta_{\mathsf{rel}}(t, Q')$. It is easy to see that $t \in Q(D)$ if and only if there exists a set U valid for (Q', D', k, F_{MM}, B), along the same lines as the argument given above for $\mathsf{QRD}(\text{FO}, F_{\mathsf{MS}})$.

(2.2) Upper bound. We next provide a PSPACE algorithm for $\mathsf{QRD}(\text{FO}, F)$, when F is F_{MS} or F_{MM}. The algorithm works as follows:

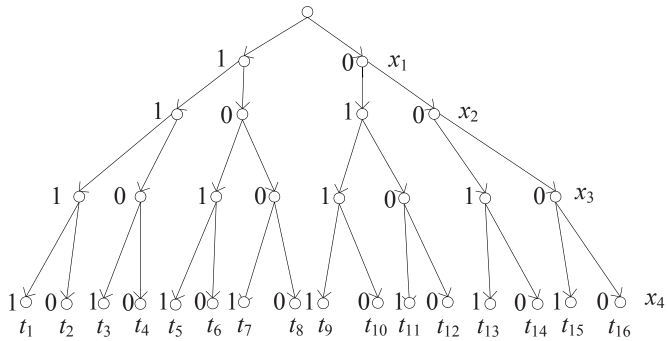
1. guess a set U consisting of k distinct tuples of R_Q;
2. check whether $U \subseteq Q(D)$ and $F(U) \geq B$; if so, return "yes", and otherwise reject the guess and go back to step 1.

The algorithm is in NPSPACE since step 2 is in PSPACE. Indeed, checking whether $U \subseteq Q(D)$ is in PSPACE when Q is an FO query, and checking $F(U) \geq B$ is in PTIME when F is F_{MS} or F_{MM}. Thus the algorithm is in NPSPACE = PSPACE. As a result, $\mathsf{QRD}(\text{FO}, F_{\mathsf{MS}})$ and $\mathsf{QRD}(\text{FO}, F_{\mathsf{MM}})$ are in PSPACE. □

We next establish the combined complexity of $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{mono}})$

THEOREM 5.2. *The combined complexity of* $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{mono}})$ *is* PSPACE-*complete, no matter whether $\mathcal{L}_Q$ is CQ, UCQ, $\exists$FO$^+$or FO.* □

In contrast to its counterparts for F_{MS} and F_{MM}, the lower bound proof for F_{mono} needs a counting argument and is more involved. It is verified by reduction from Q3SAT: given a sentence $\varphi = P_1 x_1 \ldots P_m x_m \psi$, where P_i is either $\forall$ or $\exists$ and ψ is an



$$\varphi = \exists x_1 \forall x_2 \exists x_3 \forall x_4 \psi,\ \psi = (x_1 \vee x_2 \vee \bar{x}_3) \wedge (\bar{x}_2 \vee \bar{x}_3 \vee x_4).$$

$l = 3, P_4 = \forall:$

$$\begin{gathered}
\delta_{\mathsf{dis}}(t_1, t_2) = 0 \Leftarrow \psi[t_1/\vec{x}] = 1,\, \psi[t_2/\vec{x}] = 0,\\
\delta_{\mathsf{dis}}(t_3, t_4) = 1 \Leftarrow \psi[t_3/\vec{x}] = 1, \psi[t_4/\vec{x}] = 1,\\
\delta_{\mathsf{dis}}(t_5, t_6) = 1 \Leftarrow \psi[t_5/\vec{x}] = 1,\, \psi[t_6/\vec{x}] = 1,\\
\delta_{\mathsf{dis}}(t_7, t_8) = 1 \Leftarrow \psi[t_7/\vec{x}] = 1,\, \psi[t_8/\vec{x}] = 1,\\
\delta_{\mathsf{dis}}(t_9, t_{10}) = 0 \Leftarrow \psi[t_9/\vec{x}] = 1,\, \psi[t_{10}/\vec{x}] = 0,\\
\delta_{\mathsf{dis}}(t_{11}, t_{12}) = 1 \Leftarrow \psi[t_{11}/\vec{x}] = 1,\, \psi[t_{12}/\vec{x}] = 1,\\
\delta_{\mathsf{dis}}(t_{13}, t_{14}) = 0 \Leftarrow \psi[t_{13}/\vec{x}] = 0,\, \psi[t_{14}/\vec{x}] = 0,\\
\delta_{\mathsf{dis}}(t_{15}, t_{16}) = 1 \Leftarrow \psi[t_{15}/\vec{x}] = 1,\, \psi[t_{16}/\vec{x}] = 1.
\end{gathered}$$

$l = 2, P_3 = \exists:$

$$\begin{gathered}
\delta_{\mathsf{dis}}(t_i, t_j) = 1,\, i \in [1, 2], j \in [3, 4] \Leftarrow \delta_{\mathsf{dis}}(t_1, t_2) = 0,\, \delta_{\mathsf{dis}}(t_3, t_4) = 1,\\
\delta_{\mathsf{dis}}(t_i, t_j) = 1,\, i \in [5, 6], j \in [7, 8] \Leftarrow \delta_{\mathsf{dis}}(t_5, t_6) = 1,\, \delta_{\mathsf{dis}}(t_7, t_8) = 1,\\
\delta_{\mathsf{dis}}(t_i, t_j) = 1,\, i \in [9, 10], j \in [11, 12] \Leftarrow \delta_{\mathsf{dis}}(t_9, t_{10}) = 0,\, \delta_{\mathsf{dis}}(t_{11}, t_{12}) = 1,\\
\delta_{\mathsf{dis}}(t_i, t_j) = 1,\, i \in [13, 14], j \in [15, 16] \Leftarrow \delta_{\mathsf{dis}}(t_{13}, t_{14}) = 0,\, \delta_{\mathsf{dis}}(t_{15}, t_{16}) = 1.
\end{gathered}$$

$l = 1, P_2 = \forall:$

$$\begin{gathered}
\delta_{\mathsf{dis}}(t_i, t_j) = 1,\, i \in [1, 4], j \in [5, 8] \Leftarrow \delta_{\mathsf{dis}}(t_1, t_4) = 1,\, \delta_{\mathsf{dis}}(t_5, t_8) = 1,\\
\delta_{\mathsf{dis}}(t_i, t_j) = 1,\, i \in [9, 12], j \in [13, 16] \Leftarrow \delta_{\mathsf{dis}}(t_9, t_{12}) = 1,\, \delta_{\mathsf{dis}}(t_{13}.t_{16}) = 1.
\end{gathered}$$

$l = 0, P_1 = \exists:$

$$\delta_{\mathsf{dis}}(t_i, t_j) = 1,\, i \in [1, 8], j \in [9, 16] \Leftarrow \delta_{\mathsf{dis}}(t_1, t_8) = 1,\, \delta_{\mathsf{dis}}(t_9, t_{16}) = 1.$$

Fig. 2. Example distance function δ_{dis} when $m = 4$ in the lower bound proof of Theorem 5.2

instance of 3SAT over variables in $X = \{x_1, \ldots, x_m\}$, Q3SAT is to decide whether φ is true. It is known to be PSPACE-complete (cf. [Papadimitriou 1994]).

To give the reduction, we need some notations and a lemma. Consider an instance φ of Q3SAT. We want to use "Boolean" tuples $t = (u_1, \ldots, u_m)$ to encode a true assignment for variables in X of φ, where u_i is either 0 or 1 for $i \in [1, m]$. Denote by t^l the *prefixes* $(u_1, \ldots, u_l)$ of length l, for $l \in [0, m-1]$. Consider a pair of tuples $t = (u_1, \ldots, u_m)$ and $s = (v_1, \ldots, v_m)$ such that $t^l = s^l$ but $u_{l+1} \neq v_{l+1}$, *i.e.,* t and s agree on the first l attribute values but differ in the $(l+1)$-th attribute. Note that t^l and s^l define a truth assignment of variables x_i for $i \in [1, l]$, referred to as *the truth assignment encoded by* t^l. In the reduction, we want to define a distance function δ_{dis} such that $\delta_{\mathsf{dis}}(t, s) = 1$ if and only if formula $P_{l+1}x_{l+1} \ldots P_m x_m \psi$ is satisfied by the truth assignment encoded by prefix t^l. More specifically, we define δ_{dis} inductively from $l = m-1$ down to $l = 0$.

(i) When $l = m-1$, *i.e.,* t and s differ only in their last attribute, we define $\delta_{\mathsf{dis}}(t, s)= 1$ if either (a) $P_m = \forall$ and both the truth assignments encoded by t an s make ψ true, or (b) $P_m = \exists$, and there exists at least one truth assignment μ_X (encoded by either t or s) such that μ_X satisfies ψ. For any other tuples t' and s', we define $\delta_{\mathsf{dis}}(t', s') = 0$.

(ii) When $m-2 \geq l \geq 0$, we define $\delta_{\mathsf{dis}}(t, s) = 1$ (a) when $P_{l+1} = \forall$, and $\delta_{\mathsf{dis}}((t^l, 1, 1, \ldots, 1), (t^l, 1, 0, \ldots, 0)) = 1$ and $\delta_{\mathsf{dis}}((s^l, 0, 1 \ldots, 1), (s^l, 0, 0, \ldots, 0)) = 1$; or (b) when $P_{l+1} = \exists$, and at least one of $\delta_{\mathsf{dis}}((t^l, 1, 1, \ldots, 1), (t^l, 1, 0, \ldots, 0))$ and $\delta_{\mathsf{dis}}((s^l, 0, 1 \ldots, 1), (s^l, 0, 0, \ldots, 0))$ is 1. For any other tuples t' and s', we define $\delta_{\mathsf{dis}}(t', s') = 0$.

An example δ_{dis} is depicted in Fig. 2, for a sentence φ with 4 variables.

Such distance functions have the following property. That is, the cases of t^{l+1} and s^{l+1} (branches of truth assignments t^l) given above suffice to ensure Lemma 5.3.

LEMMA 5.3. *Consider any pair of Boolean tuples $t = (u_1, \ldots, u_m)$ and $s = (v_1, \ldots, v_m)$ that encode truth assignments of $\varphi = P_{l+1}x_{l+1} \ldots P_m x_m \psi$. If $t^l = s^l$ and $u_{l+1} \neq v_{l+1}$ for some $0 \geq l \geq m-1$, then $\delta_{\mathsf{dis}}(t, s) = 1$ if and only if φ is true under μ_X^l, where μ_X^l is the truth assignment for variables $x_1, \ldots, x_l$ encoded by t^l.* □

PROOF. We prove Lemma 5.3 by induction on l from $l = m-1$ down to $l = 0$. When $l = m-1$, by the definition of δ_{dis}, we have that $\delta_{\mathsf{dis}}((t^{m-1}, 1), (t^{m-1}, 0)) = 1$ if and only if (a) when $P_m = \forall$, the truth assignments encoded by $(t^{m-1}, 1)$ and $(t^{m-1}, 0)$ both satisfy ψ; and (b) when $P_m = \exists$, at least one of the truth assignments encoded by $(t^{m-1}, 1)$ and $(t^{m-1}, 0)$ makes ψ true. Hence the statement holds in the base case.

To give more intuition for the inductive step, we also illustrate the case when $l = m-2$. For any two tuples $t = (u_1, \ldots, u_m)$ and $s = (v_1, \ldots, v_m)$ such that $t^{m-2} = s^{m-2}$ and $u_{m-1} \neq v_{m-1}$, by the definition of δ_{dis}, we have that $\delta_{\mathsf{dis}}(t, s) = 1$ if and only if (a) when $P_{m-1} = \forall$, $\delta_{\mathsf{dis}}((t^{m-2}, 1, 1), (t^{m-2}, 1, 0)) = 1$ and $\delta_{\mathsf{dis}}((t^{m-2}, 0, 1), (t^{m-2}, 0, 0)) = 1$; and (b) when $P_{m-1} = \exists$, we have that either $\delta_{\mathsf{dis}}((t^{m-2}, 1, 1), (t^{m-2}, 1, 0)) = 1$ or $\delta_{\mathsf{dis}}((t^{m-2}, 0, 1),(t^{m-2}, 0, 0)) = 1$. Then case (a) holds if and only if (i) when $P_m = \forall$, the truth assignments encoded by $(t^{m-2}, 1, 1)$ and $(t^{m-2}, 1, 0)$ both make ψ true; similarly for $(t^{m-2}, 0, 1)$ and $(t^{m-2}, 0, 0)$; and (ii) when $P_m = \exists$, there exists at least one truth assignment (encoded by $(t^{m-2}, 1, 1)$ or $(t^{m-2}, 1, 0)$) that satisfies ψ; similarly for $(t^{m-2}, 0, 1)$ and $(t^{m-2}, 0, 0)$. Clearly, case (a) (resp. case (b)) holds if and only if $\forall x_{m-1} P_m x_m\ \psi$ (resp. $\exists x_{m-1} P_m x_m\ \psi$) is true under the truth assignment encoded by t^{m-2}.

Now we prove the inductive step. Assume that when $l = p$ and $p \in [1, m-3]$, Lemma 5.3 holds. That is, for all tuples $t = (u_1, \ldots, u_m)$ and $s = (v_1, \ldots, v_n)$ such that $t^p = s^p$ but $u_{p+1} \neq v_{p+1}$, $\delta_{\mathsf{dis}}(t,\ s) = 1$ if and only if $P_{p+1}x_{p+1} \ldots P_m x_m\ \psi$ is true under the truth assignment encoded by t^p for variables $x_1, \ldots, x_p$. We next consider the case when $l = p-1$. Let t and s be an arbitrary pair of tuples that agree on the first $p-1$ attributes but disagree on the p-th attribute. By the definition of δ_{dis}, we have that $\delta_{\mathsf{dis}}(t, s) = 1$ if and only if (c) when $P_p = \forall$, $\delta_{\mathsf{dis}}((t^{p-1}, 1, 1, \ldots, 1), (t^{p-1}, 1, 0, \ldots, 0)) = 1$ and $\delta_{\mathsf{dis}}((t^{p-1}, 0, 1, \ldots, 1), (t^{p-1}, 0, 0, \ldots, 0)) = 1$; and (d) when $P_p = \exists$, either $\delta_{\mathsf{dis}}((t^{p-1}, 1, 1, \ldots, 1), (t^{p-1}, 1, 0, \ldots, 0)) = 1$ or $\delta_{\mathsf{dis}}((t^{p-1}, 0, 1, \ldots, 1), (t^{p-1}, 0, 0, \ldots, 0)) = 1$. By the induction hypothesis, for $l = p$, we know that case (c) holds if and only if $P_{p+1}x_{p+1} \ldots P_m x_m\ \ \psi$ is true under the truth assignments of variables $x_1, \ldots, x_{p-1}$ encoded by $\vec{u}_{p-1}$, no matter whether x_p is 1 or 0. The latter holds if and only if $\forall x_p P_{p+1} x_{p+1} \ldots P_m x_m\ \psi$ is true under the truth assignment of $x_1, \ldots, x_{p-1}$ encoded by t^{p-1}. Similarly, case (d) is satisfied if and only if $\exists x_p P_{p+1} x_{p+1} \ldots P_m x_m\ \psi$ is true under the truth assignment of $x_1, \ldots, x_{p-1}$ encoded by t^{p-1}. This proves Lemma 5.3. □

Leveraging Lemma 5.3, we next prove Theorem 5.2.

PROOF. It suffices to show that $\mathsf{QRD}(\mathcal{CQ}, F_{\mathsf{mono}})$ is PSPACE-hard, even when $\lambda = 1$ and k is a constant, and that $\mathsf{QRD}(\mathcal{FO}, F_{\mathsf{mono}})$ is in PSPACE.

(1) Lower bound. We show that $\mathsf{QRD}(\mathcal{CQ}, F_{\mathsf{mono}})$ is PSPACE-hard by reduction from the Q3SAT problem. Given an instance $\varphi = P_1 x_1 \ldots P_m x_m \psi$ of Q3SAT with variables in $X = \{x_1, \ldots, x_m\}$, we construct a CQ query Q, a database D, functions δ_{rel}, δ_{dis} and F_{mono}, and let $k = 1$ and $B = 1$. We prove that φ is true if and only if there exists a valid set U for $(Q, D, k, F_{\mathsf{mono}}, B)$.

(1) Database D has a single unary relation $I_{01} = \{(1), (0)\}$ specified by schema $R_{01}(X)$, encoding the Boolean domain as in the proof of Theorem 5.1 for the FO case.

(2) We define the CQ query Q as follows:

$$Q(\vec{x}) = R_{01}(x_1) \wedge \ldots \wedge R_{01}(x_m),$$

where $\vec{x} = (x_1, \ldots, x_m)$. Intuitively, query Q generates all truth assignments for variables in X. Note that $|Q(D)| = 2^m$.

(3) The function δ_{rel} is a constant function that returns 1 for each tuple of R_Q. We use the distance function δ_{dis} given above, and set $\lambda = 1$.

We next verify that there exists a valid set U for $(Q, D, k, F_{\mathsf{mono}}, B)$ if and only if φ is true, by giving a counting argument.

$\boxed{\Rightarrow}$ Assume that there exists a valid set $U = \{t\}$ for $(Q, D, k = 1, F_{\mathsf{mono}}, B = 1)$. Then by $\lambda = 1$, $F_{\mathsf{mono}}(U) = \frac{1}{2^m - 1} \cdot \sum_{s \in Q(D)} \delta_{\mathsf{dis}}(t, s) \geq 1$. Thus $\sum_{s \in Q(D)} \delta_{\mathsf{dis}}(t, s) \geq 2^m - 1$. Since $|Q(D)| = 2^m$, for each tuple $s \in Q(D)$, if $s \neq t$, $\delta_{\mathsf{dis}}(t, s)$ must be 1 by the definition of δ_{dis}. In particular, there exists a tuple s' such that s' differs from t in the first attribute and $\delta_{\mathsf{dis}}(t, s') = 1$. Then by Lemma 5.3, we have that $\varphi = P_1 x_1 \ldots P_m x_m \psi$ is true.

$\boxed{\Leftarrow}$ Conversely, assume that φ is true. We show that there must exist a valid set U. By Lemma 5.3, for each pair of tuples t and s such that $t^1 \neq s^1$, we have that $\delta_{\mathsf{dis}}(t, s) = 1$. Moreover, since φ is true, no matter what P_1 is, there must exist a truth assignment μ_{x_1} for variable x_1 such that $P_2 x_2 \ldots P_m x_m \psi$ is true under μ_{x_1}. Then again by Lemma 5.3, for each pair of tuples $t' = (u'_1, \ldots, u'_m)$ and $s' = (v'_1, \ldots, v'_m)$, $\delta_{\mathsf{dis}}(t', s') = 1$ if $t'^1 = s'^1 = \mu_{x_1}$ but $u'_2 \neq v'_2$. Furthermore, since $P_2 x_2 \ldots P_m x_m \psi$ is true under the truth assignment μ_{x_1} for variable x_1, there must exist a truth assignment μ_{x_2} of variable x_2 such that $P_3 x_3 \ldots P_m x_m \psi$ is true under μ_{x_1} and μ_{x_2}. Similarly, we get truth assignments μ_{x_l} for variable x_l, where $l \in [3, m]$, such that for each pair of tuples t and s, $\delta_{\mathsf{dis}}(t, s) = 1$, if $t^l = s^l = (\mu_{x_1}, \mu_{x_2}, \ldots, \mu_{x_l})$, and $u_{l+1} \neq v_{l+1}$ for $l \in [0, m-1]$. Let $t^* = (\mu_{x_1}, \ldots, \mu_{x_m})$ and $U = \{t^*\}$. Then by the definition of Q, $t^* \in Q(D)$. Obviously, $|U| = 1$. To see that $F_{\mathsf{mono}}(U) = \frac{1}{2^m - 1} \sum_{s \in Q(D)} \delta_{\mathsf{dis}}(t^*, s) \geq B = 1$, we compute the number of tuples s such that $\delta_{\mathsf{dis}}(t^*, s) = 1$. As discussed earlier, for each tuple $s = (v_1, \ldots, v_m)$, $\delta_{\mathsf{dis}}(t^*, s) = 1$ if there exists $l \in [0, m-1]$ such that $(t^*)^l = s^l$ but $\mu_{x_{l+1}} \neq v_{l+1}$. It is easy to see that there are $2^m / 2^{l+1} = 2^{m-l-1}$ tuples s such that $\delta_{\mathsf{dis}}(t^*, s) = 1$. Thus the number of tuples s such that $\delta_{\mathsf{dis}}(t^*, s) = 1$ is $\sum_{l=0}^{m-1} 2^{m-l-1} = 2^{m-1} + \ldots + 2 + 1 = 2^m - 1$. Recall that we set $k = 1$ and $B = 1$. Hence $F_{\mathsf{mono}}(U) = \frac{1}{2^m - 1} \sum_{s \in Q(D)} \delta_{\mathsf{dis}}(t^*, s) = 1 \geq B$.

(2) Upper bound. We show that $\mathsf{QRD}(\mathcal{FO}, F_{\mathsf{mono}})$ is in PSPACE. Let M_Q be the binary representation of tuples t of R_Q such that each of its attributes has the largest constants from the active domain. Note that tuple t is bounded by $\mathsf{arity}(R_Q)$ and $\mathsf{adom}(Q, D)$, where $\mathsf{arity}(R_Q)$ denotes the arity of the schema R_Q, and $\mathsf{adom}(Q, D)$ is the set of constants appearing in D or Q. We present an NPSPACE algorithm as follows:

1. guess a set U of k tuples of R_Q;
2. if $U \subseteq Q(D)$, continue; otherwise, reject the guess and go back to step 1;

3. let n denote the number of tuples in $Q(D)$, and variable dist keep track of the sum of distances between each pair of tuples $t \in Q(D)$ and $s \in Q(D)$, both represented in binary; initially, we set $n = 0$ and $\mathsf{dist} = 0$;
4. for each tuple $s \in [0, M_Q]$ (encoded in binary with necessary delimiters) do:
   a. check whether $s \in Q(D)$; if so, continue; otherwise, go back to step 4;
   b. for each tuple $t \in U$, $\mathsf{dist} := \mathsf{dist} + \delta_{\mathsf{dis}}(t, s)$;
   c. increase n by 1;
5. compute $F_{\mathsf{mono}}(U) = (1 - \lambda) \cdot \sum_{t \in U} \delta_{\mathsf{rel}}(t, Q) + (\lambda/(n + 1)) \cdot \mathsf{dist}$; check whether $F_{\mathsf{mono}}(U) \geq B$; if so, return "yes".

Observe the following. (a) Step 2 is in PSPACE for FO queries. (b) Steps 4 and 5 can also be done in polynomial space when tuples t, counter n and sum dist are encoded in binary. Indeed, step 4(a) is in PSPACE for FO queries. Moreover, steps 4(b), 4(c) and 5 can be done in PSPACE because we consider tuples in binary coding. Hence the algorithm is in NPSPACE = PSPACE. Thus $\mathsf{QRD}(\mathrm{FO}, F_{\mathsf{mono}})$ is in PSPACE. □

5.2. The Data Complexity of QRD

We next re-investigate QRD for data complexity. That is, when query Q is predefined and fixed, but database D may vary, *i.e.,* the complexity of evaluating a *fixed* query for variable database inputs (see [Abiteboul et al. 1995] for details). Data complexity is also of practical interest since in many real-life applications, one often uses a fixed set of queries, *e.g.,* Web forms, while the database may be frequently updated.

From the result below we can see that when the objective is given by F_{MS} and F_{MM}, fixing query Q does not reduce the complexity of QRD for CQ, UCQ and $\exists\mathrm{FO}^+$: the problem remains NP-complete, the same as its combined complexity. In contrast, fixed queries do simplify the analysis of the problem.

(1) when $\mathcal{L}_Q$ is FO and F is F_{MS} or F_{MM}, $\mathsf{QRD}(\mathcal{L}_Q, F)$ becomes NP-complete; or

(2) when F is F_{mono}, the problem becomes tractable in this case.

Contrast these with the PSPACE-completeness of their combined complexity (Theorem 5.1 and 5.2). These demonstrate that when Q is fixed, objective functions determine the data complexity, while query languages *have no impact*.

THEOREM 5.4. *The data complexity of* $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{MS}})$ *and* $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{MM}})$ *is* NP-*complete, while that of* $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{mono}})$ *is in* PTIME*, when $\mathcal{L}_Q$ is CQ, UCQ, $\exists$FO$^+$or FO.* □

PROOF. We first study the data complexity of $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{MS}})$ and $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{MM}})$ for CQ, UCQ, $\exists\mathrm{FO}^+$and FO, and then investigate it of $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{mono}})$.

<u>(1) When F is F_{MS} or F_{MM}.</u> It suffices to prove that $\mathsf{QRD}(\mathrm{CQ}, F)$ is NP-hard and that $\mathsf{QRD}(\mathrm{FO}, F)$ is in NP. Recall that the lower bounds of $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MS}})$ and $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MM}})$ of Theorem 5.1 are shown by using a fixed identity query. Thus the lower bounds hold here, *i.e.,* $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MS}})$ and $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MM}})$ are NP-hard. For the upper bound, note that the algorithm given in the proof of Theorem 5.1 for FO can carry over here. Clearly, its step 2 is in PTIME since $Q(D)$ is PTIME computable for a fixed FO query Q, and F is also in PTIME when F is F_{MS} or F_{MM}. Thus the algorithm is in NP for fixed FO queries, and hence so is the data complexity of $\mathsf{QRD}(\mathrm{FO}, F_{\mathsf{MS}})$ and $\mathsf{QRD}(\mathrm{FO}, F_{\mathsf{MM}})$.

<u>(2) When F is F_{mono}.</u> We give a PTIME algorithm to check whether there exists a set U valid for $(Q, D, k, F_{\mathsf{mono}}, B)$. Given a fixed Q, D, F_{mono}, k and B, the algorithm first computes the entire set $Q(D)$ of query answers. It then checks whether $Q(D)$ contains at least k items at all, and if so, it picks k tuples from $Q(D)$ with the largest "score" in terms of the relevance and diversity. Finally, it checks whether the sum of the scores of these tuples is beyond the bound B. It returns "yes" if so and "no" otherwise. When Q

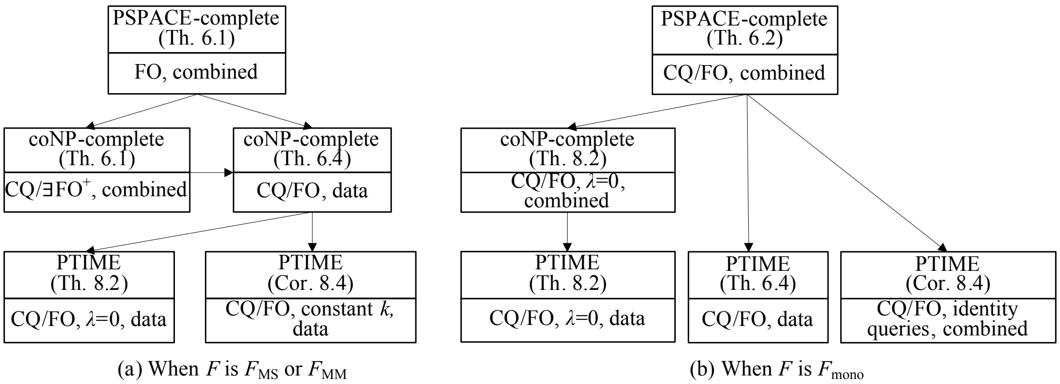


Fig. 3. The complexity bounds of DRP

is fixed, all these can be done in PTIME. In contrast to F_{MS} and F_{MM}, F_{mono} allows us to compute the scores by inspecting each tuple in $Q(D)$ individually, rather than by checking each k-element subset of $Q(D)$. More specifically, the algorithm works as follows:

1. compute $Q(D)$, in PTIME;
2. check whether $|Q(D)| \geq k$; if so, continue; otherwise return "no";
3. for each tuple t, compute $v(t) = (1-\lambda)\cdot\delta_{\mathsf{rel}}(t,Q)+\frac{\lambda}{|Q(D)|-1}\cdot\sum_{s\in Q(D)}\delta_{\mathsf{dis}}(t,s)$ in PTIME;
4. let v_{sum} be the sum of the k largest values in U, where $U = \{v(t) \mid t \in Q(D)\}$, and check whether $v_{\mathsf{sum}} \geq B$; if so, return "yes"; otherwise, return "no".

The algorithm is in PTIME. Indeed, since Q is fixed, $Q(D)$ is PTIME computable. Thus steps 1-3 are all in PTIME. Hence so is the data complexity of QRD($\mathbf{FO}, F_{\mathsf{mono}}$). □

6. THE DIVERSITY RANKING PROBLEM

We next study the decision problem DRP. We give its combined complexity in Section 6.1 and data complexity in Section 6.2, and will investigate its special cases in Section 8. Along the same lines as Fig. 1, we summarize the complexity bounds of DRP in various setting in Fig. 3, which depicts the impact of various parameters.

All the complexity bounds of this section remain intact when rank r is taken as input of DRP instead of a constant, except the PTIME bound of the data complexity for F_{mono} (Theorem 6.4; see details there).

6.1. The Combined Complexity of DRP

While DRP does not reduce to QRD (or vice versa), the two decision problems are connected. Given an instance D, Q, F, k and $U \subseteq Q(D)$ of DRP, one can construct an instance D, Q, F, k and $B = F(U)$ of QRD such that $\mathsf{rank}(U) \leq r$ if and only if there exist no more $r-1$ subsets U' of $Q(D)$ with $F(U') > B$. From this an algorithm for DRP immediately follows, by using an QRD oracle: first guess r subsets U' of $Q(D)$ with k elements each, and then check whether $F(U') > B$ by calling (a minor revision of) a procedure for QRD; return "no" if so. In light of the connection, it is not surprising that DRP and QRD behave similarly in their combined complexity analyses.

(1) When F is F_{MS} or F_{MM}, $\mathcal{L}_Q$ dominates the combined complexity of DRP.

(2) When F is F_{mono}, $\mathcal{L}_Q$ has no impact on the combined complexity of DRP.

We first study the combined complexity of DRP($\mathcal{L}_Q, F_{\mathsf{MS}}$) and DRP($\mathcal{L}_Q, F_{\mathsf{MM}}$).

THEOREM 6.1. *The combined complexity of* DRP($\mathcal{L}_Q, F_{\mathsf{MS}}$) *and* DRP($\mathcal{L}_Q, F_{\mathsf{MM}}$) *is*

— coNP-*complete when* $\mathcal{L}_Q$ *is CQ, UCQ or* $\exists$*FO*$^+$*, and*
— PSPACE-*complete when* $\mathcal{L}_Q$ *is FO.* □

The proof is similar to the proof of Theorem 5.1 for QRD. In particular, the lower bounds are verified by reductions from the complements of 3SAT and the membership problem for FO. Note that, however, the connection between QRD to DRP is not strong enough for us to carry (the dual of) the complexity bounds of QRD to DRP: (1) DRP does not inherit the lower bound of QRD since QRD does not reduce to DRP, and (2) for the upper bound, the algorithm outlined above gives us Σ_2^p by calling the QRD, not coNP. Here Σ_2^p is the class of languages recognized by a nondeterministic Turing machine with an NP oracle, *i.e.,* $\mathsf{NP}^{\mathsf{NP}}$ [Papadimitriou 1994].

PROOF. We first show that $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{MS}})$ and $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{MM}})$ are coNP-complete when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$, and then prove that they are PSPACE-complete for FO.

(1) When $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$. It suffices to show that $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{MS}})$ and $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{MM}})$ are coNP-hard for CQ and that they are in coNP for $\exists$FO$^+$.

(1.1) Lower bound. We show that $\mathsf{DRP}(\mathrm{CQ}, F_{\mathsf{MS}})$ and $\mathsf{DRP}(\mathrm{CQ}, F_{\mathsf{MM}})$ are already coNP-hard for fixed identity queries, $\lambda = 1$ and $r = 1$, by reductions from the complement of 3SAT. We first verify this for $\mathsf{DRP}(\mathrm{CQ}, F_{\mathsf{MS}})$. Given an instance φ of 3SAT, where $\varphi = C_1 \wedge \ldots \wedge C_l$ is defined over variables in $X = \{x_1, \ldots, x_m\}$, we define a database D, a CQ query Q, functions δ_{rel}, δ_{dis} and F_{MS}, a set U and a positive integer k. We show that φ is not satisfiable if and only if $\mathsf{rank}(U) \leq r$.

To give the reduction, we construct a new formula φ' such that φ' is satisfiable if and only if φ is satisfiable. Let z be a fresh variable that is not in the set X of variables in φ. We define $\varphi' = (\varphi \vee z) \wedge \bar{z} = \bigwedge_{i=1}^{l} (C_i \vee z) \wedge \bar{z}$. It is easy to see that for a truth assignment μ_X of X variables, μ_X satisfies φ if and only if μ_X makes φ' true with $z = 0$. Moreover, there always exists a truth assignment that makes φ' false. Indeed, when setting z to be 1, φ' is false under any truth assignments of X. We next give the reduction.

(1) The database D includes a single relation I_C specified by the following schema: $R_C(\mathsf{cid}, L_1, V_1, L_2, V_2, L_3, V_3, Z, V_Z, A)$, populated as follows. Let $C_i' = l_1^i \vee l_2^i \vee l_3^i \vee z$ be the i-th clause of φ', where $i \in [1, l]$. For any possible truth assignment μ_i of variables in the literals of C_i', we add a tuple $(i, x_k, v_k, x_l, v_l, x_m, v_m, z, v_z, a)$, where $x_k = l_1^i$ if $l_1^i \in X$, and $x_k = \bar{l}_1^i$ if $l_1^i = \bar{x}_k$. We set $v_k = \mu_i(x_k)$; similarly for x_l, x_m, z and v_l, v_m and v_z. Moreover, we set $a = 1$ if the truth assignment μ_i satisfies C_i'; otherwise we set $a = 0$. Further, for $\bar{z}$, we add two tuples $(l+1, e_1, f_1, e_2, f_2, e_3, f_3, z, 1, 0)$ and $(l+1, e_1, f_1, e_2, f_2, e_3, f_3, z, 0, 1)$, where all e_i and f_i are distinct constants that are not in $X \cup \{z, 0, 1\}$.

(2) We define Q as the identity query on R_C instances.

(3) Let $k = l + 1$. We let the set U consist of $l + 1$ tuples from D, one for each clause in φ' such that all variables in X and z are set to be 1. Obviously, $U \subseteq Q(D)$.

(4) We define the relevance function δ_{rel} to be a constant function that returns 1 for each tuple t of R_Q. Moreover, for each pair of tuples t and s of R_Q, we define $\delta_{\mathsf{dis}}(t, s) = 1$ if (i) $t[\mathsf{cid}] \neq s[\mathsf{cid}]$, *i.e.,* t and s encode two different clauses in φ'; (ii) t and s have the same value for each variable appearing in both t and s; and (iii) $t[A] = s[A] = 1$, *i.e.,* t and s encode truth assignments that satisfy their corresponding clauses, respectively. Furthermore, for any other pair of tuples t' and s', we define $\delta_{\mathsf{dis}}(t', s') = 0$. Moreover, we let $\lambda = 1$. Then for each set S of tuples of R_Q, $F_{\mathsf{MS}}(S) = \sum_{t,s \in S} \delta_{\mathsf{dis}}(t, s)$.

We next show that $\mathsf{rank}(U) = 1 \leq r$ if and only if φ is not satisfiable.

$\boxed{\Rightarrow}$ Assume that φ is satisfiable. Then there exists a truth assignment μ_X^0 for the X variables that satisfies φ. We show that $\mathsf{rank}(U) \geq 2 > r = 1$. Let U^0 consist of $l + 1$

tuples, one for each clause in φ', such that the values of all variables in X agree with μ_X^0 and z is set to be 0. Obviously, for any two different tuples t and s in U^0, we have that $\delta_{\mathsf{dis}}(t,s) = 1$ by the definition of δ_{dis}. Then $F_{\mathsf{MS}}(U^0) = (l+1)\cdot l$. Note that for each tuple $t \in U$, $t[A] = 1$ if $t \neq (l+1, e_1, f_1, e_2, f_2, e_3, f_3, z, 1, 0)$, and moreover, for any two tuples $t, s \in U$ such that $t[A] = s[A] = 1$, we have that $\delta_{\mathsf{dis}}(t,s) = 1$, by the definition of δ_{dis}. Then $F_{\mathsf{MS}}(U) = l \cdot (l-1)$. Putting these together, we have that $\mathsf{rank}(U) \geq 2 > r = 1$.

$\boxed{\Leftarrow}$ Conversely, assume that φ is not satisfiable. Then there exists no truth assignment μ_X of the X variables that satisfies φ. It is easy to see that for each candidate set S for (Q, D, k), there exist at most l tuples $t \in S$ such that $t[A] = 1$, and thus $F_{\mathsf{MS}}(S) \leq l \cdot (l-1) = F_{\mathsf{MS}}(U)$. Therefore, $\mathsf{rank}(U) = 1 \leq r = 1$.

We show that $\mathsf{DRP}(\mathtt{CQ}, F_{\mathsf{MM}})$ is also coNP-hard by reduction from the complement of 3SAT. Given an instance φ of 3SAT, we construct the same φ', D, Q, k, r, $\delta_{\mathsf{rel}}(\cdot,\cdot)$, U and $\lambda = 1$ as above. We define distance function δ'_{dis} such that for any pair of distinct tuples t and s of R_Q, (i) $\delta'_{\mathsf{dis}}(t,s) = 2$ if $\delta_{\mathsf{dis}}(t,s) = 1$ and $t, s \notin U$; (ii) $\delta'_{\mathsf{dis}}(t,s) = 1$ if $t, s \in U$; and (iii) for any other t' and s', $\delta_{\mathsf{dis}}(t', s') = 0$. Then for each k-element set S of tuples of schema R_Q, $F_{\mathsf{MM}}(S) = \min_{t,s\in S, t\neq s} \delta_{\mathsf{dis}}(t,s)$. We show that this is indeed a reduction. By the definitions of δ'_{dis} and F_{MM}, for each candidate set S for (Q, D, k), (i) $F_{\mathsf{MM}}(S) = 2$ if S encodes a truth assignment of X that satisfies φ with $z = 0$; (ii) $F_{\mathsf{MM}}(S) = 1$ if $S = U$; and otherwise (iii) $F_{\mathsf{MM}}(S) = 0$. Then along the same line as for $\mathsf{DRP}(\mathtt{CQ}, F_{\mathsf{MS}})$, one can verify that φ is not satisfiable if and only if $\mathsf{rank}(U) = 1 \leq r = 1$.

(1.2) Upper bound. We show that when F is F_{MS} or F_{MM}, $\mathsf{DRP}(\exists\mathtt{FO}^+, F)$ is in coNP, by giving an NP algorithm for its complement. Given Q, D, F, U and k, the algorithm checks whether $\mathsf{rank}(U) > r$, *i.e.,* whether there exist r candidate sets for (Q, D, k) such that for each such set S, $F(S) > F(U)$.

1. make r guess such that each time, guess k CQ queries from Q, and for each CQ query, guess a tableau from D; these tableaux yield a subset of $Q(D)$, collected in $\mathcal{S}$;
2. check whether $\mathcal{S}$ contains r distinct sets and whether for each $S \in \mathcal{S}$, $|S| = k$;
3. for each set $S \in \mathcal{S}$, check if $F(S) > F(U)$; if so, return "no".

The algorithm is in NP since step 2 is in PTIME; moreover, step 3 is also in PTIME since F is PTIME computable when F is F_{MS} or F_{MM}. Hence the problem is in coNP.

(2) When $\mathcal{L}_Q$ is FO. We show that DRP is PSPACE-complete for F_{MS} and F_{MM}.

(2.1) Lower bound. We show that for FO, DRP is already PSPACE-hard when $\lambda = 0$ and $r = 1$, by reductions from the complement of the membership problem for FO.

We first consider $\mathsf{DRP}(\mathtt{FO}, F_{\mathsf{MS}})$. Given an instance (Q, D, s) of the membership problem for FO, we define a database $D' = (D, I_{01})$, where $I_{01} = \{(1), (0)\}$ is an instance of schema $R_{01}(X)$ encoding the Boolean domain. We define a query Q' as follows:

$$Q'(\vec{x}, z, c) = \big(Q(\vec{x}) \vee (R_{01}(z) \wedge z = 1)\big) \wedge R_{01}(c).$$

Clearly, no matter whether $s \in Q(D)$ or not, $(s,1,1)$ and $(s,1,0)$ are in $Q'(D')$. Moreover, when $s \in Q(D)$, $(s,0,1)$ and $(s,0,0)$ must be in $Q'(D')$. We define $\delta_{\mathsf{rel}}((s,0,1), Q') = \delta_{\mathsf{rel}}((s,0,0), Q') = 3$, $\delta_{\mathsf{rel}}((s,1,1), Q') = \delta_{\mathsf{rel}}((s,1,0), Q') = 2$, and for any other tuple t of $R_{Q'}$, $\delta_{\mathsf{rel}}(t, Q') = 1$. Furthermore, we define δ_{dis} as a constant function that returns 0 for each pair of tuples of R_Q. We set $\lambda = 0$ and $k = 2$; hence, for each set S of tuples of R'_Q with k tuples, $F_{\mathsf{MS}}(S) = (k-1)\cdot \sum_{t\in S} \delta_{\mathsf{rel}}(t, Q')$. Note that $F_{\mathsf{MS}}(\{(s,0,1), (s,0,0)\}) = 6$, $F_{\mathsf{MS}}(\{(s,1,1), (s,1,0)\}) = 4$, $F_{\mathsf{MS}}(\{t, t'\}) = 5$ if $t \in \{(s,0,1), (s,0,0)\}$ and $t' \in \{(s,1,1), (s,1,0)\}$, and for any other pair of tuples t and t', $F_{\mathsf{MS}}(\{t, t'\}) = 2$. Finally, let $r = 1$ and $U = \{(s,1,1), (s,1,0)\}$. As remarked earlier, $U \subseteq Q(D)$.

We show that $s \notin Q(D)$ if and only if $\mathsf{rank}(U) \leq r$. First assume that $s \notin Q(D)$. Then $(s, 0, 1)$ and $(s, 0, 0)$ are not in $Q'(D')$. Thus $\mathsf{rank}(U) = 1 \leq r$ by the definition of F_{MS}. Conversely, if $s \in Q(D)$, $(s, 0, 1)$ and $(s, 0, 0)$ are both in $Q'(D')$, and $\mathsf{rank}(U) > r = 1$.

We next show that $\mathsf{DRP}(\mathrm{FO}, F_{\mathsf{MM}})$ is PSPACE-hard, also by reduction from the complement of the membership problem for FO. Given an instance (Q, D, s) of that problem, we define the same Q', D', δ_{rel}, $\lambda = 0$ and $r = 1$ as above. Then for each set S of tuples of R'_Q, $F_{\mathsf{MM}}(S) = \min_{t \in S} \delta_{\mathsf{rel}}(t, Q')$. Let $U = \{(s, 1, 1)\}$. Then along the same line as above, one can verify that $s \notin Q(D)$ if and only if $\mathsf{rank}(U) = 1 \leq r$ for $(Q', D', k, F_{\mathsf{MM}})$.

(2.2) Upper bound. We next present a PSPACE algorithm for the complement of $\mathsf{DRP}(\mathrm{FO}, F)$ when F is F_{MS} or F_{MM}. The algorithm works as follows:

1. guess r distinct sets such that each such set S consists of k different tuples of R_Q;
2. denote by $\mathcal{S}$ the collection of the r sets; for each $S \in \mathcal{S}$, check whether $S \subseteq Q(D)$; if so, continue;
3. check whether for all $S \in \mathcal{S}$, $F(S) > F(U)$; if so, return "no".

The algorithm is in NPSPACE = PSPACE since step 2 is in PSPACE for FO queries and step 3 is in PTIME since F_{MS} or F_{MM} are in PTIME; hence so is the problem. □

We next study the combined complexity of $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{mono}})$

THEOREM 6.2. *The combined complexity of* $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{mono}})$ *is* PSPACE-*complete when* $\mathcal{L}_Q$ *is CQ, UCQ,* $\exists$*FO*$^+$*or FO.* □

The proof is a variation of the one for Theorem 5.2. The lower bound is also verified by reduction from Q3SAT. Given an instance $\varphi = P_1 x_1 \ldots P_m x_m \psi$ of Q3SAT over variables $X = \{x_1, \ldots, x_m\}$, we define a distance function δ^*_{dis} similar to its counterpart given for QRD. More specifically, let $t = (u_1, \ldots, u_m)$ and $s = (v_1, \ldots, v_m)$ be any pair of tuples encoding two truth assignments of X variables, such that $t^l = s^l$ but $u_{l+1} \neq v_{l+1}$, where t^l and s^l are prefixes of t and s of length l, respectively. We want to use δ^*_{dis} to ensure that $\delta^*_{\mathsf{dis}}(t, s) > 0$ if and only if formula $P_{l+1} x_{l+1} \ldots P_m x_m \psi$ is satisfied by the truth assignment encoded by prefix t^l, for variables $x_1, \ldots, x_l$. To do so, denote by $\hat{t}$ the m-arity tuple $(1, \ldots, 1)$. We define δ^*_{dis} by revising δ_{dis} given in the proof of Theorem 5.2 for $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{mono}})$, such that

(i) $\delta^*_{\mathsf{dis}}(\hat{t}, s) = (1/2) \cdot \delta_{\mathsf{dis}}(\hat{t}, s)$ for all tuples $s = (1, v_2, \ldots, v_m)$ with $v_i \in \{0, 1\}$ for $i \in [2, m]$;

(ii) $\delta^*_{\mathsf{dis}}(\hat{t}, s) = 2 \cdot \delta_{\mathsf{dis}}(\hat{t}, s)$ for all $s = (0, v_2, \ldots, v_m)$ with $v_i \in \{0, 1\}$ for $i \in [2, m]$; and

(iii) for any other pair of tuples t' and s' of R_Q, $\delta^*_{\mathsf{dis}}(t', s') = \delta_{\mathsf{dis}}(t', s')$.

It is easy to see that for each pair t and s of m-arity tuples given above, $\delta_{\mathsf{dis}}(t, s) = 1$ if and only if $\delta^*_{\mathsf{dis}}(t, s) > 0$. Along the same line as Lemma 5.3, one can show the following.

LEMMA 6.3. *Consider any pair of tuples* $t = (u_1, \ldots, u_m)$ *and* $s = (v_1, \ldots, v_m)$ *that encode truth assignments of* $\varphi = P_{l+1} x_{l+1} \ldots P_m x_m \psi$*, where* $t^l = s^l$ *and* $u_{l+1} \neq v_{l+1}$ *for some* $0 \geq l \geq m - 1$*. Let* μ^l_X *be a truth assignment of variables* $x_1, \ldots, x_l$ *encoded by* t^l*. Then the following statements are equivalent: (1)* $P_{l+1} x_{l+1} \ldots P_m x_m \psi$ *is satisfied by* μ^l_X*, (2)* $\delta^*_{\mathsf{dis}}(t, s) > 0$*, and (3) there exist two tuples* $t' = (u'_1, \ldots, u'_m)$ *and* $s' = (v'_1, \ldots, v'_m)$ *for* φ *such that* $\delta^*_{\mathsf{dis}}(t', s') > 0$*, where* $t'^l = s'^l = \mu^l_X$ *but* $u'_{l+1} \neq v'_{l+1}$*.* □

Capitalizing on Lemma 6.3, we next prove Theorem 6.2.

PROOF. We show that $\mathsf{DRP}(\mathrm{CQ}, F_{\mathsf{mono}})$ is PSPACE-hard $\mathsf{DRP}(\mathrm{FO}, F_{\mathsf{mono}})$is in PSPACE. The lower bound holds even when $\lambda = 1$ and k is a constant.

(1) Lower bound. We show the lower bound by reduction from Q3SAT. Given an instance $\varphi = P_1x_1 \ldots P_mx_m\psi$ of Q3SAT, we construct the same query Q, database D and relevance function δ_{rel} as their counterparts for QRD(CQ, F_{mono}), and let $k = 1$ and $r = 1$. Furthermore, let $U = \{\hat{t}\}$, where $\hat{t}$ is the m-arity tuple $(1, \ldots, 1)$ in $Q(D)$ given above. Finally, we set $\lambda = 1$. Then for each set S of tuples of R_Q, $F_{\mathsf{mono}}(S) = \frac{1}{2^m-1}\sum_{t\in S, s\in Q(D)} \delta^*_{\mathsf{dis}}(t,s)$, where δ^*_{dis} is defined as above.

We show that φ is true if and only if $\mathsf{rank}(U) = 1 \leq r = 1$.

$\boxed{\Rightarrow}$ Assume first that φ is true. Then by Lemma 6.3 we have that $\delta^*_{\mathsf{dis}}(\hat{t}, s) = 2$ for each tuple $s = (0, v_2, \ldots, v_m)$, where $v_i \in \{0, 1\}$ for $i \in [2, m]$. Note that there exist 2^{m-1} such tuples s in total. Thus $\sum_{s\in Q(D)} \delta^*_{\mathsf{dis}}(\hat{t}, s) \geq 2 \cdot 2^{m-1} = 2^m$. We next prove that for any other tuple t that is distinct from $\hat{t}$, we have that $\sum_{s\in Q(D)} \delta^*_{\mathsf{dis}}(t, s) \leq 2^m$, and thus $\mathsf{rank}(U) = 1$ by the definition of F_{mono}. Indeed, for any tuple $t \neq \hat{t}$, there exist at most $2^m - 1$ tuples $s \neq t$ such that $\delta^*_{\mathsf{dis}}(t, s) > 0$ (recall that $|Q(D)| = 2^m$). Consider the following two cases. For any tuple $t = (u_1, \ldots, u_m)$ and $t \neq \hat{t}$, (a) if $u_1 = 0$, then by the definition of δ^*_{dis}, $\sum_{s\in Q(D)} \delta^*_{\mathsf{dis}}(t, s) = \sum_{s\in Q(D)\setminus\{\hat{t}\}} \delta^*_{\mathsf{dis}}(t, s) + \delta^*_{\mathsf{dis}}(t, \hat{t}) \leq (2^m-2)+2 = 2^m$; and (b) otherwise, $\sum_{s\in Q(D)} \delta^*_{\mathsf{dis}}(t, s) = \sum_{s\in Q(D)\setminus\{\hat{t}\}} \delta^*_{\mathsf{dis}}(t, s)+\delta^*_{\mathsf{dis}}(t, \hat{t}) \leq (2^m - 2) + 1/2 = 2^m - 3/2 < 2^m$. As a result, we have that $\mathsf{rank}(U) = 1 \leq r = 1$.

$\boxed{\Leftarrow}$ Conversely, assume that φ is false. Let l_0 be the minimum value in $[0, m - 1]$ such that there exists a tuple $s = (v_1, \ldots, v_m)$ with $\delta_{\mathsf{dis}}(\hat{t}, s) = 1$, where $s^{l_0} = (1, \ldots, 1)$ and $v_{l_0+1} \neq 1$. Then by Lemma 6.3, l_0 is also the minimum value in $[0, m - 1]$ such that $\delta_{\mathsf{dis}}(t', s') = 1$ for any pair of tuples $t' = (u'_1, \ldots, u'_m)$ and $s' = (v'_1, \ldots, v'_m)$, where $t'^{l_0} = s'^{l_0} = (1, \ldots, 1)$ but $u'_{l_0+1} \neq v'_{l_0+1}$. Note that there exist at most $2^{m-l_0} - 1$ tuples s such that $\delta^*_{\mathsf{dis}}(\hat{t}, s) > 0$, where $s^{l_0} = (1, \ldots, 1)$ and $v_{l_0+1} \neq 1$ (recall that there are at most 2^{m-l_0} tuples s such that $s^{l_0} = (1, \ldots, 1)$ and $v_{l_0+1} \neq 1$). Then we have that $\sum_{s\in Q(D)} \delta_{\mathsf{dis}}(\hat{t}, s) \leq (1/2)\cdot (2^{m-l_0} - 1) = 2^{m-l_0-1} - 1/2$. We next show that there exists a tuple t^* that is distinct from $\hat{t}$ such that $\sum_{s\in Q(D)} \delta_{\mathsf{dis}}(t^*, s) \geq 2^{m-l_0-1}$, and hence $\mathsf{rank}(U) > r = 1$. Indeed, let $t^* = (u^*_1, \ldots, u^*_m)$ be a tuple such that $(t^*)^{l_0-1} = (1, \ldots, 1)$ but $u^*_{l_0} \neq 1$. Then there exist at least 2^{m-l_0-1} tuples $s = (v_1, \ldots, v_m)$ such that $\delta_{\mathsf{dis}}(t^*, s) > 0$, where $(t^*)^{l_0} = s^{l_0}$ but $u^*_{l_0+1} \neq v_{l_0+1}$, and moreover, for each such tuple s, $\delta^*_{\mathsf{dis}}(t^*, s) = 1$ by the definition of δ^*_{dis}. Thus we have that $\sum_{s\in Q(D)} \delta_{\mathsf{dis}}(t^*, s) \geq 2^{m-l_0-1}$. Hence by the definition of F_{mono}, $F_{\mathsf{mono}}(U) < F_{\mathsf{mono}}(U')$, for each set U' consisting of a single tuple t^* given above. Hence $\mathsf{rank}(U) > r = 1$.

(2) Upper bound. We show that DRP(FO, F_{mono}) is in PSPACE. Recall the algorithm given earlier for DRP(FO, F) for F_{MS} and F_{MM}. Obviously, the algorithm also works here. We show that it is in PSPACE. Indeed, since F_{mono} is PSPACE computable (as argued in the proof of Theorem 5.2), step 3 of that algorithm is in PSPACE here; moreover, step 2 is in PSPACE. Thus the algorithm is in NPSPACE = PSPACE. □

6.2. The Data Complexity of DRP

One might be tempted to think that fixing Q would make DRP($\mathcal{L}_Q, F$) easier. Nevertheless, DRP($\mathcal{L}_Q, F$) becomes simpler only (1) when F is F_{mono}, or (2) when $\mathcal{L}_Q$ is FO, and F is F_{MS} or F_{MM}. Its data complexity remains the same as its combined complexity when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$, for F_{MS} and F_{MM} (see Theorem 6.4). This is consistent with the data complexity analysis of QRD($\mathcal{L}_Q, F$) (Theorem 5.4).

THEOREM 6.4. *For CQ, UCQ, $\exists$FO$^+$and FO, the data complexity of* $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{MS}})$ *and* $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{MM}})$ *are* coNP*-complete, while that of* $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{mono}})$ *is in* PTIME. □

As remarked earlier, all the results of this section remain valid even when r is treated as part of the input for DRP rather than a constant, except the data complexity of DRP for F_{mono}. Indeed, if r is not a constant and if the numeric value is encoded in binary instead of unary, the PTIME algorithm to be presented below for the F_{mono} case becomes pseudo-polynomial time rather than PTIME. The other proofs of Theorem 6.4 are quite similar to their counterparts for Theorem 5.4.

PROOF. We first investigate the data complexity of DRP when F is F_{MS} or F_{MM}. We then study it when F is F_{mono}.

(1) When F is F_{MS} or F_{MM}. It suffices to prove that $\mathsf{DRP}(\mathsf{CQ}, F_{\mathsf{MS}})$ and $\mathsf{DRP}(\mathsf{CQ}, F_{\mathsf{MM}})$ are coNP-hard and that $\mathsf{DRP}(\mathsf{FO}, F_{\mathsf{MS}})$ and $\mathsf{DRP}(\mathsf{FO}, F_{\mathsf{MM}})$ are in coNP for fixed queries.

Recall that the lower bounds of $\mathsf{DRP}(\mathsf{CQ}, F_{\mathsf{MS}})$ and $\mathsf{DRP}(\mathsf{CQ}, F_{\mathsf{MM}})$ for Theorem 6.1 are established by using a fixed identity query. Thus the lower bounds hold here. Hence $\mathsf{DRP}(\mathsf{CQ}, F_{\mathsf{MS}})$ and $\mathsf{DRP}(\mathsf{CQ}, F_{\mathsf{MM}})$ are both coNP-hard. For the upper bound, the algorithm given in the proof of Theorem 6.1 for FO and for F_{MS} and F_{MM} works here, which is to check whether $\mathsf{rank}(U) > r$. We show that the algorithm is in NP. Indeed, $Q(D)$ is PTIME computable when Q is a fixed FO query. Hence its step 2 is PTIME. Moreover, step 3 is in PTIME for F_{MS} and F_{MM} since F is PTIME computable. Thus the algorithm is in NP, and as a result, $\mathsf{DRP}(\mathsf{FO}, F_{\mathsf{MS}})$ and $\mathsf{DRP}(\mathsf{FO}, F_{\mathsf{MM}})$ are in coNP.

(2) When the objective is mono-objective. We show that $\mathsf{DRP}(\mathsf{FO}, F_{\mathsf{mono}})$ is in PTIME by giving a PTIME algorithm. Given Q, D, k, r, δ_{dis}, δ_{dis}, F_{mono}, and a set U, the algorithm returns "yes" if $\mathsf{rank}(U) \leq r$, and returns "no" otherwise.

Intuitively, the algorithm first finds a collection $\mathcal{S}$ of top-r candidate sets for (Q, D, k) based on F_{mono}. Let $\mathcal{S} = \{S_1, \ldots, S_r\}$, where $F_{\mathsf{mono}}(S_1) \geq \ldots \geq F_{\mathsf{mono}}(S_r)$. Then we simply need to check whether $F_{\mathsf{mono}}(U) < F_{\mathsf{mono}}(S_r)$; the algorithm returns no if so, and yes otherwise. To see this, observe that for each candidate set $V \notin \mathcal{S}$, there exists no set $S \in \mathcal{S}$ such that $F_{\mathsf{mono}}(V) > F_{\mathsf{mono}}(S)$. After $\mathcal{S}$ is found, consider the following cases. (a) If $U \in \mathcal{S}$, then there exist at most $r - 1$ candidate sets V such that $F_{\mathsf{mono}}(V) > F_{\mathsf{mono}}(U)$, and thus $\mathsf{rank}(U) \leq r$. (b) If $U \notin \mathcal{S}$ but $F_{\mathsf{mono}}(U) = F_{\mathsf{mono}}(S_r)$, then similarly, $\mathsf{rank}(U) \leq r$. (c) If $U \notin \mathcal{S}$ and $F_{\mathsf{mono}}(U) < F_{\mathsf{mono}}(S_r)$, then $\mathsf{rank}(U) > r$ since there exist at least r candidate sets V such that $F_{\mathsf{mono}}(V) > F_{\mathsf{mono}}(U)$. Thus the algorithm checks which condition of (a), (b) or (c) is satisfied by $\mathcal{S}$ and U. It returns "yes" in both cases (a) and (b), and returns "no" in case (c).

We next show how to compute collection $\mathcal{S}$. We construct $\mathcal{S}$ by adding sets $S_1, \ldots, S_r$ to $\mathcal{S}$, one set or multiple sets at a time. Let $\mathsf{FindNext}(Q, D, F_{\mathsf{mono}}, \mathcal{S}, \mathcal{S}', k, l, l')$ be a procedure that given Q, D, F_{mono}, k, l and a collection $\mathcal{S}$ that consists of top-l candidate sets for (Q, D, k), returns a collection $\mathcal{S}'$ such that $\mathcal{S} \subset \mathcal{S}'$ and $\mathcal{S}'$ is a collection of top-l' candidate sets for (Q, D, k), where $1 \leq l < l' \leq r$, by finding one or more candidate sets $S \notin \mathcal{S}$. Procedure $\mathsf{FindNext}(Q, D, F_{\mathsf{mono}}, \mathcal{S}, \mathcal{S}', k, l, l')$ works as follows. Let $\mathcal{S} = \{S_1, \ldots, S_l\}$ $(l > 0)$. For each tuple $t \in Q(D)$, let $v(t) = (1 - \lambda) \cdot \delta_{\mathsf{rel}}(t, Q)$ $+(\lambda/(|Q(D)| - 1)) \sum_{t' \in Q(D)} \delta_{\mathsf{dis}}(t, t')$. Then it carries out the following steps.

1. For each $S_i \in \mathcal{S}$, find sets V by replacing *one* tuple t in S_i with another tuple s in $Q(D) \setminus S_i$, where $v(s) \leq v(t)$, such that sets V have the highest F_{mono} values among all such new sets. More specifically, do the following:
   a. For each $t \in S_i$ and $s \in Q(D)$ such that $s \notin S_i$ and $v(s) \leq v(t)$, we get the candidate set $S_i(s, t)$ by replacing t in S_i with s. Obviously, $F_{\mathsf{mono}}(S_i(s, t)) \leq F_{\mathsf{mono}}(S_i)$.

Denote by $E_i(t)$ the collection of such sets $S_i(s,t)$ with the highest F_{mono} value such that $S_i(s,t) \notin \mathcal{S}$. Note that $|E_i(t)| \geq 0$. When all tuples t in S_i are processed, we get k sets $E_i(t)$ for each $t \in S_i$. Let E_i be the collection of candidate sets in $\bigcup_{t\in S_i} E_i(t)$ with the highest F_{mono} value. Note that for each set $V \in E_i$, we have that $F_{\mathsf{mono}}(V) \leq F_{\mathsf{mono}}(S_i)$ and $V \notin \mathcal{S}$.

b. Let E be the collection of sets in $E_1 \cup \ldots \cup E_l$ with the highest F_{mono} value. We will show that $\mathcal{S} \cup E$ must consist of the top-l' candidate sets. Note that $|\mathcal{S} \cup E|$ may be greater than r while $|\mathcal{S}| < r$.

c. Check whether $|\mathcal{S} \cup E| > r$. If so, we get $\mathcal{S}'$ by adding only $r - |\mathcal{S}|$ sets from E to $\mathcal{S}$, picked randomly as they have the same F value; otherwise, let $\mathcal{S}'$ be $\mathcal{S} \cup E$.

2. Return $\mathcal{S}'$.

Capitalizing on $\mathsf{FindNext}()$, the algorithm for DRP with fixed FO queries is given as follows, which returns "yes" if $\mathsf{rank}(U) \leq r$, and returns "no" otherwise:

1. compute $Q(D)$; for each $t \in Q(D)$, compute $v(t) = (1-\lambda)\cdot\delta_{\mathsf{rel}}(t,Q) + (\lambda/(|Q(D)|-1)) \sum_{t'\in Q(D)} \delta_{\mathsf{dis}}(t,t')$; sort tuples in $Q(D)$ in descending order based on their $v()$ values;
2. let $\mathcal{S}$ be the collection consisting of top-l candidate sets for (Q,D,k); initially, $\mathcal{S}$ contains only the set S_1 that consists of the first k tuples in the sorted $Q(D)$; obviously, $\mathcal{S}$ is the top-1 candidate set; moreover, let $l = 1$;
3. while $|\mathcal{S}| < r$, do the following:
   a. let $\mathcal{S}'$ = $\mathsf{FindNext}(Q, D, F_{\mathsf{mono}}, \mathcal{S}, \mathcal{S}', k, l, l')$;
   b. let $\mathcal{S}$ be $\mathcal{S}'$;
4. when $|\mathcal{S}| = r$, check whether condition (a) or (b) given above is satisfied by $\mathcal{S}$ and U; if so, return "yes"; otherwise return "no".

We show that the algorithm is correct and is in PTIME. Obviously it is correct if and only if $\mathsf{FindNext}(Q, D, F_{\mathsf{mono}}, \mathcal{S}, \mathcal{S}', k, l, l')$ finds top-l' candidate sets. Assume *w.l.o.g.* that $|Q(D)| > k$. We show that FindNext finds top-l' candidate sets by induction on l such that when FindNext finds top-l' candidate sets $\mathcal{S}'$ based on the top-l candidate sets $\mathcal{S}$ (where $1 < l < l'$), then FindNext can find the top-l'' candidate sets $\mathcal{S}''$ based on $\mathcal{S}'$ for some $l'' > l'$. Obviously, we need only to consider the case when $l'' \leq r$, *i.e.,* $\mathcal{S}''$ contains all the new candidate sets with the highest F_{mono} value found by FindNext.

When $l = 1$, $\mathcal{S}$ contains only the set S_1 that consists of the first k tuples in the sorted $Q(D)$. Based on $\mathcal{S}$, FindNext finds all sets V by replacing tuples $t \in S_1$ *one at a time* with a tuple $s \in Q(D)\setminus S_1$, where $v(s) \leq v(t)$. Denote by E the collection consisting of all such sets V that have the highest F_{mono} value. Let $\mathcal{S}' = \mathcal{S} \cup E$. We show that for any candidate set $V' \notin \mathcal{S}'$, $F_{\mathsf{mono}}(V') \leq F_{\mathsf{mono}}(S)$ for each set $S \in \mathcal{S}'$. Note that $F_{\mathsf{mono}}(V') \leq F_{\mathsf{mono}}(S_1)$ since S_1 is the top-1 candidate set. Now we only need to prove that $F_{\mathsf{mono}}(V') \leq F_{\mathsf{mono}}(S)$ for each set $S \in E$. Consider the following two cases. (a) There exist tuples $t \in S_1$ and $s \in Q(D) \setminus S_1$ such that $v(s) \leq v(t)$ and V' is obtained by replacing t in S_1 with s. Recall that $V' \notin E$ since $V' \notin \mathcal{S}'$. Thus we have that $F_{\mathsf{mono}}(V') \leq F_{\mathsf{mono}}(S)$ for each set $S \in E$, since all sets in E have the highest F_{mono} value in sets obtained by an one-tuple replacement. (b) The set V' is not one given in (a) above. Then V' must contain no more than $k-2$ tuples in S_1. Assume that V' is obtained by replacing tuples $t_1, \ldots, t_j$ in S_1 with $s_1 \ldots, s_j$, where $j \in [2,k]$, such that $s_1, \ldots, s_j$ are not in S_1 and for each $i \in [1,j]$, $v(s_i) \leq v(t_i)$. Let V'' be the set obtained by replacing only one tuple t_1 in S_1 with s_1. Obviously, $F_{\mathsf{mono}}(V') \leq F_{\mathsf{mono}}(V'')$. Moreover, we have that $F_{\mathsf{mono}}(V'') \leq F_{\mathsf{mono}}(S)$ for each set $S \in E$, since all the sets in E have the highest F_{mono} value. Thus $F_{\mathsf{mono}}(V') \leq F_{\mathsf{mono}}(S)$ for each $S \in E$. Hence $\mathcal{S}'$ contains the top-$(1+|E|)$ candidate sets.

Assume that when $l = n$ and $n \in [2,r]$, FindNext finds the top-l' candidate sets $\mathcal{S}'$ based on the top-l candidate sets $\mathcal{S}$ for some $l' > l$. For the inductive step, we consider the case when $l = l'$. Let the collection E' consist of all sets V with the highest F_{mono}

value found by FindNext based on the top-l' candidate sets $\mathcal{S}'$, following the one-tuple-at-a-time replacement strategy. Let $\mathcal{S}'' = \mathcal{S}' \cup E'$. We show that $\mathcal{S}''$ contains the top-$(l' + |E'|)$ candidate sets. It suffices to prove that for each candidate set $V' \subseteq Q(D)$ such that $V' \notin \mathcal{S}''$, $F_{\mathsf{mono}}(V') \leq F_{\mathsf{mono}}(S)$ for each $S \in \mathcal{S}''$. Note that by the induction hypothesis, for each set $S \in \mathcal{S}'$, $F_{\mathsf{mono}}(V') \leq F_{\mathsf{mono}}(S)$ since $\mathcal{S}'$ contains the top-l' candidate sets. Thus we need only to show that $F_{\mathsf{mono}}(V') \leq F_{\mathsf{mono}}(S)$ for each set $S \in E'$. This can be verified along the same lines as the argument for expanding $l = 1$ above, examining cases (a) and (b) given there. This verifies the correctness of the algorithm.

We next show that the algorithm is in PTIME. Note that procedure FindNext is called at most $r - 1$ times. In each call, for each set S in $\mathcal{S}$, there are at most $k \cdot |Q(D)|$ replacements of tuples in S. Thus, at most $O(r \cdot k \cdot |Q(D)|)$ time is needed for each call of FindNext. Putting these together, the algorithm is in $O((r - 1) \cdot r \cdot k \cdot |Q(D)|)$ time. Since Q is fixed, r is a constant and $Q(D)$ is bounded by a polynomial in $|D|$, the algorithm is in PTIME. Therefore, the data complexity of $\mathsf{DRP}(\mathsf{FO}, F_{\mathsf{mono}})$ is in PTIME.

We remark that the algorithm given above is in pseudo-polynomial time if r is not a constant and if r is encoded in binary. It is in PTIME when k is a constant. □

7. THE RESULT DIVERSITY COUNTING PROBLEM

We now study the counting problem RDC. We establish its combined complexity in Section 7.1 and data complexity in Section 7.2; we will also identify and investigate its special cases in Section 8. Along the same lines as Fig. 1, we depict in Fig. 4 the connections between complexity bounds of RDC in various settings.

7.1. The Combined Complexity of RDC

We first study the combined complexity of RDC. Here we use the framework of predicate-based counting classes introduced in [Hemaspaandra and Vollmer 1995]. For a complexity class C of decision problems, $\#{\cdot}\mathsf{C}$ is the class of all counting problems associated with a predicate R_L that satisfies the following conditions:

— R_L is polynomially balanced, *i.e.,* there exists a polynomial q such that for all strings x and y, if $R_L(x, y)$ is true then $|y| \leq q(|x|)$; and
— the following decision problem is in class C: "given x and y, it is to decide whether $R_L(x, y)$".

A counting problem is to compute the cardinality of the set $\{y \mid R_L(x, y)\}$, *i.e.,* it is to find how many y there are such that predicate $R_L(x, y)$ is satisfied.

We show that when the objective is for max-sum or max-min diversification, the problem becomes harder for FO than for CQ, UCQ and $\exists$FO$^+$. In contrast, when the objective is for mono-objective formulation, F_{mono} has greater impact on the complexity than $\mathcal{L}_Q$: it remains $\#{\cdot}$PSPACE-complete when $\mathcal{L}_Q$ ranges over CQ, UCQ, $\exists$FO$^+$and FO. This is consistent with its counterparts for QRD and DRP.

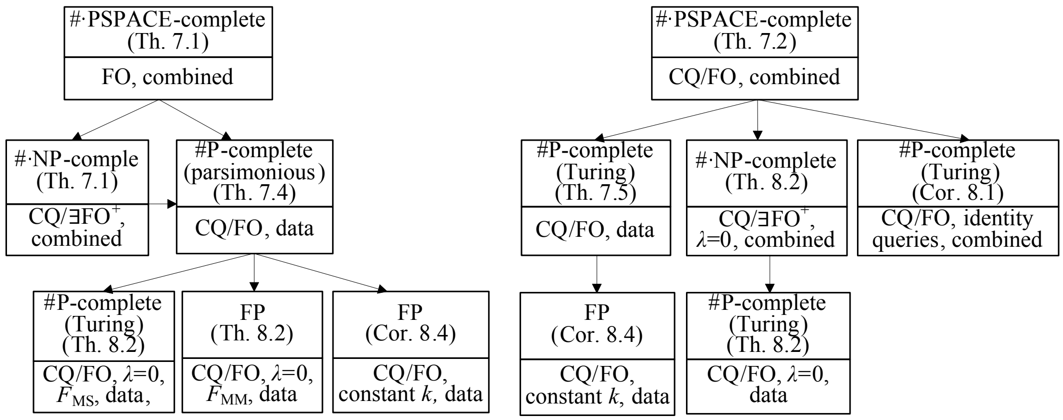
The results are verified by parsimonious reductions. A *parsimonious reduction* from a counting problem $\#A$ to a counting problem $\#B$ is a PTIME function σ such that for all x, $|\{y \mid (x, y) \in A\}| = |\{z \mid (\sigma(x), z) \in B\}|$, *i.e.,* σ is a bijection [Durand et al. 2005].

Below we first study $\mathsf{RDC}(\mathcal{L}_Q, F)$ when F is F_{MS} or F_{MM}.

THEOREM 7.1. *The combined complexity of* $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{MS}})$ *and* $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{MM}})$ *is*

— $\#{\cdot}$NP-*complete when* $\mathcal{L}_Q$ *is CQ, UCQ or* $\exists$*FO*$^+$*, and*
— $\#{\cdot}$PSPACE-*complete when* $\mathcal{L}_Q$ *is FO.*

All the results hold under parsimonious reductions. □



(a) When F is F_{MS} or F_{MM} (b) When F is F_{mono}

Fig. 4. The complexity bounds of RDC

$$I_{01} = \begin{array}{|c|}\hline X \\ \hline 1 \\ 0 \end{array} \qquad I_{\vee} = \begin{array}{|ccc|}\hline B & A_1 & A_2 \\ \hline 0 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{array} \qquad I_{\wedge} = \begin{array}{|ccc|}\hline B & A_1 & A_2 \\ \hline 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{array} \qquad I_{\neg} = \begin{array}{|cc|}\hline A & \bar{A} \\ \hline 0 & 1 \\ 1 & 0 \end{array}$$

Fig. 5. Relation instances used in the lower bound proofs of Theorem 7.1.

The lower bounds are verified by reductions from the following problems.

(1) $\#\Sigma_1$SAT: Given an existentially quantified Boolean formula of the form $\varphi(X, Y) = \exists X \psi(X, Y)$, where $\psi(X, Y)$ is of the form $C_1 \wedge \ldots \wedge C_l$, and C_i is disjunctions of variables or negated variables taken from $X = \{x_1, \ldots, x_m\}$ and $Y = \{y_1, \ldots, y_n\}$, it is to count the number of truth assignments of Y that satisfy φ. It is known that $\#\Sigma_1$SAT is $\#\cdot$NP-complete [Durand et al. 2005].

(2) #QBF: Given a Boolean formula of the form $\varphi = \exists X \forall y_1 P_2 y_2 \cdots P_n y_n \psi$, where $P_i \in \{\exists, \forall\}$ for $i \in [2, n]$, and ψ is quantifier-free Boolean formula over the variables in $X = \{x_1, \ldots, x_m\}$ and $Y = \{y_1, \ldots, y_n\}$, it is to count the number of truth assignments of X variables that satisfy φ. It is known to be $\#\cdot$PSPACE-complete [Ladner 1989].

Given these, we prove Theorem 7.1 as follows.

PROOF. We start with RDC for CQ, UCQ and $\exists$FO$^+$, and then study them for FO.

(1) When $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$. It suffices to show that RDC(CQ, F_{MS}) and RDC(CQ, F_{MM}) are $\#\cdot$NP-hard and that RDC($\exists$FO$^+$, F_{MS}) and RDC($\exists$FO$^+$, F_{MM}) are in $\#\cdot$NP.

(1.1) Lower bound. We first show that RDC(CQ, F_{MS}) is $\#\cdot$NP-hard even when $\lambda = 0$ and k is a constant, by parsimonious reductions from $\#\Sigma_1$SAT. Given an instance $\varphi(X, Y) = \exists X \psi(X, Y)$ of $\#\Sigma_1$SAT, we define a database D, a CQ query Q, functions δ_{rel}, δ_{dis} and F_{MS}, a positive integer k and a real number B. We show that the number of valid sets for $(Q, D, k, F_{\text{MS}}, B)$ equals the number of truth assignments of Y variables that satisfy φ. In particular, we set $k = 2$ and $B = 3$.

(1) The database consists of four relations I_{01}, $I_{\vee}$, $I_{\wedge}$ and $I_{\neg}$ as shown in Fig. 5, specified by schemas $R_{01}(X)$, $R_{\vee}(B, A_1, A_2)$, $R_{\wedge}(B, A_1, A_2)$ and $R_{\neg}(A, \bar{A})$, respectively. Here

I_{01} encodes the Boolean domain, and $I_\vee$, $I_\wedge$ and $I_\neg$ encode disjunction, conjunction and negation, respectively, such that we can express φ' below in CQ with these relations.

(2) To define the CQ query Q, we first construct a new formula φ' as follows:

$$\varphi'(\vec{y}) = \exists \vec{x}, z\ \big((\psi(\vec{x}, \vec{y}) \vee z) \wedge \bar{z}\big),$$

where z is a new variable not in $X \cup Y$. It can be verified that a truth assignment μ_Y of Y variables satisfies φ if and only if μ_Y makes φ' true when z is set to be 0.

We now define the CQ query Q as follows:

$$Q(\vec{y}, z, a) = \exists \vec{x}\ \big(Q_X(\vec{x}) \wedge Q_Y(\vec{y}) \wedge R_{01}(z) \wedge Q_1(\vec{x}, \vec{y}, z, a)\big).$$

Here $\vec{x} = (x_1, \ldots, x_m)$ and $\vec{y} = (y_1, \ldots, y_n)$. Queries Q_X and Q_Y generate all truth assignments of variables in X and Y, respectively, by means of Cartesian products of R_{01}. Sub-query Q_1 leverages $R_\vee$, $R_\wedge$ and $R_\neg$ to encode the formula φ'. The semantics of Q_1 is that for a given truth assignment μ_X of variables in X, μ_Y for Y and μ_z for z, which are encoded by tuples t_X, t_Y and t_Z, respectively, $Q_1(t_X, t_Y, t_z, a)$ returns $a = 1$ if $(\psi \vee z) \wedge \bar{z}$ is satisfied by μ_X, μ_Y and μ_z; and it returns $a = 0$ otherwise. Intuitively, Q returns all tuples (t_Y, t_z, a) such that Q_1 returns $a = 1$ if the truth assignment μ_Y of Y variables (encoded by t_Y) and μ_z of z (represented by t_z) satisfy φ'.

(3) We define (a) $\delta_{\mathsf{rel}}((t_Y, 0, 1), Q) = 1$, (b) $\delta_{\mathsf{rel}}((1, \ldots, 1, 0), Q) = 2$ and (c) $\delta_{\mathsf{rel}}(t, Q) = 0$ for any other tuple t of R_Q. Furthermore, we define δ_{dis} as a constant function that returns 0 for each pair of tuples of R_Q, and let $\lambda = 0$. Then for each set U of tuples of R_Q with k tuples, $F_{\mathsf{MS}}(U) = (k-1) \cdot \sum_{t \in U} \delta_{\mathsf{rel}}(t, Q)$.

We show that the construction above is indeed a parsimonious reduction. To see this, observe that for each set $U \subseteq Q(D)$ such that $|U| = 2$, $F_{\mathsf{MS}}(U) \geq 3$ if and only if $U = \{(t_Y, 0, 1), (1, \ldots, 1, 0)\}$, by the definition of F_{MS} and δ_{rel} given above, where t_Y encodes a truth assignment of Y variable that satisfies φ' with $z = 0$. Moreover, as discussed earlier, a truth assignment μ_Y makes φ true if and only if μ_Y satisfies φ' when $z = 0$. Thus, the number of truth assignments of Y variables that satisfy φ equals the number of valid sets U for $(Q, D, k, F_{\mathsf{MS}}, B)$.

We next show that $\mathsf{RDC}(\mathtt{CQ}, F_{\mathsf{MM}})$ is $\#\cdot\mathsf{NP}$-hard, also by parsimonious reduction from $\#\Sigma_1\mathsf{SAT}$. Given an instance $\varphi(X, Y) = \exists X \psi(X, Y)$ of $\#\Sigma_1\mathsf{SAT}$, we define the same query φ', database D, query Q and function δ_{dis} as given above. Furthermore, we define $\delta_{\mathsf{rel}}((t_Y, 0, 1), Q) = 1$ and for any other tuple t of R_Q, $\delta_{\mathsf{rel}}(t, Q) = 0$. Let $\lambda = 0$ and $k = 1$. Then for each set U of tuples of R_Q, $F_{\mathsf{MM}}(U) = \min_{t \in U} \delta_{\mathsf{rel}}(t, Q)$. Finally, we set $B = 1$.

We show that this also makes a parsimonious reduction. For each set $U \subseteq Q(D)$ such that $|U| = k = 1$ and $F_{\mathsf{MM}}(U) \geq B = 1$, U consists of a single tuple $(t_Y, 0, 1)$ such that t_Y encodes a truth assignment of Y variables that satisfies φ. So the number of valid sets for $(Q, D, k, F_{\mathsf{MM}}, B)$ equals the number of truth assignments of Y that satisfies φ.

(1.2) Upper bound. We show that $\mathsf{RDC}(\exists\mathtt{FO}^+, F)$ is in $\#\cdot\mathsf{NP}$, when F is F_{MS} or F_{MM}. It suffices to show that it is in NP to verify whether a given set U is valid for (Q, D, k, F, B), by the definition of $\#\cdot\mathsf{NP}$. Indeed, given a set U, it is in NP to check whether $U \subseteq Q(D)$ for $\exists\mathtt{FO}^+$, in PTIME to check whether $|U| = k$, and in PTIME to check whether $F(U) \geq B$ since F is PTIME computable when F is F_{MS} or F_{MM}. Thus $\mathsf{RDC}(\exists\mathtt{FO}^+, F)$ is in $\#\cdot\mathsf{NP}$.

(2) When $\mathcal{L}_Q$ is FO. We next study $\mathsf{RDC}(\mathtt{FO}, F_{\mathsf{MS}})$ and $\mathsf{RDC}(\mathtt{FO}, F_{\mathsf{MM}})$.

(2.1) Lower bound. We verify that $\mathsf{RDC}(\mathtt{FO}, F_{\mathsf{MS}})$ and $\mathsf{RDC}(\mathtt{FO}, F_{\mathsf{MM}})$ are $\#\cdot\mathsf{PSPACE}$-hard even when $\lambda = 0$ and k is a constant, by parsimonious reductions from #QBF.

We start with $\mathsf{RDC}(\mathtt{FO}, F_{\mathsf{MS}})$. Given an instance φ of #QBF as described above, we construct a database D, a query Q, and functions δ_{rel}, δ_{dis} and F_{MS}, and set $k = 2$ and

$B = 3$. We show that the number of valid sets for $(D, Q, k, F_{\mathsf{MS}}, B)$ equals the number of truth assignments of X that satisfies φ.

(1) The database consists of the four relations I_{01}, $I_\vee$, $I_\wedge$ and $I_\neg$ shown in Fig. 5, specified by schemas $R_{01}(X)$, $R_\vee(B, A_1, A_2)$, $R_\wedge(B, A_1, A_2)$ and $R_\neg(A, \bar{A})$, respectively.

(2) To define the query Q, we construct a new formula φ' from φ as before:

$$\varphi' = \exists X\, \forall y_1 P_2 y_2 \cdots P_n y_n \big((\psi \vee z) \wedge \bar{z}\big).$$

where z is a new variable not in $X \cup Y$. A truth assignment μ_X of X satisfies φ if and only if μ_X make φ' true with $z = 0$. Let $\psi' = (\psi \vee z) \wedge \bar{z}$. We define the query Q as follows:

$$Q(\vec{x}, z, b) = \forall y_1 P_2 y_2 \cdots P_n y_n \left(Q_X(\vec{x}) \wedge Q_Y(\vec{y}) \wedge R_{01}(z) \wedge Q_{\psi'}(\vec{x}, \vec{y}, z, b)\right).$$

Here $\vec{x} = (x_1, \ldots, x_m)$ and $\vec{y} = (y_1, \ldots, y_n)$. Queries Q_X and Q_Y generate all truth assignments of variables in X and Y, respectively, by means of Cartesian products of R_{01}. Furthermore, query $Q_{\psi'}(\vec{x}, \vec{y}, z, b)$ encodes the truth value of ψ' for given truth assignments μ_X of X variables, μ_Y of Y variables and μ_z of variable z, such that it returns $b = 1$ if ψ' is true under μ_X, μ_Y and μ_z; otherwise it returns $b = 0$. Intuitively, Q returns all tuples (t_X, t_z, b) such that Q returns $b = 1$ if the truth assignments μ_X (encoded by t_X) and μ_z (encoded by t_z) satisfy φ'; and Q returns $b = 0$ otherwise.

(3) We define (a) $\delta_{\mathsf{rel}}((t_X, 0, 1), Q) = 1$, (b) $\delta_{\mathsf{rel}}((1, \ldots, 1, 0), Q) = 2$ and (c) for any other tuple t of R_Q, $\delta_{\mathsf{rel}}(t, Q) = 0$. Furthermore, we define δ_{dis} as a constant function that returns 0 for each pair of tuples of R_Q. Finally, we set $\lambda = 0$. Then for each set U of tuples of R_Q with k tuples, $F_{\mathsf{MS}}(U) = (k-1) \cdot \sum_{t \in U} \delta_{\mathsf{rel}}(t, Q)$.

Then along the same line as the proof of $\mathsf{RDC}(\mathbf{CQ}, F_{\mathsf{MS}})$ given earlier, one can readily verify that the number of truth assignments of X variables that satisfy φ is equal to the number of sets U valid for $(Q, D, F_{\mathsf{MS}}, k, B)$.

We now show that $\mathsf{RDC}(\mathbf{FO}, F_{\mathsf{MM}})$ is $\#{\cdot}\mathsf{PSPACE}$-hard, also by parsimonious reduction from $\#\mathbf{QBF}$. Given an instance φ of $\#\mathbf{QBF}$, we construct the same formula φ', database D, query Q, and function δ_{dis} as above. Moreover, we define $\delta_{\mathsf{rel}}((t_X, 0, 1), Q) = 1$ for each m-arity tuple t_X that encodes a truth assignment of X variables, and $\delta_{\mathsf{rel}}(t, Q) = 0$ for any other tuple t of R_Q. Let $\lambda = 0$, $k = 1$ and $B = 1$. Then for each set U of consisting of k tuples of R_Q, $F_{\mathsf{MM}}(U) = \min_{t \in U} \delta_{\mathsf{rel}}(t, Q)$. Then following the proof for $\mathsf{RDC}(\mathbf{FO}, F_{\mathsf{MS}})$, one can verify that the number of truth assignments of X variables that satisfy φ equals the number of sets U valid for $(Q, D, F_{\mathsf{MM}}, k, B)$.

(2.2) Upper bound. To verify that $\mathsf{RDC}(\mathbf{FO}, F)$ is in $\#{\cdot}\mathsf{PSPACE}$ for F_{MS} and F_{MM}, we only need to show that verifying whether a given set U is valid is in PSPACE. Indeed, given a set U, it is in PSPACE to check whether $U \subseteq Q(D)$ and it is in PTIME to check whether $|U| = k$ and $F(U) \geq B$ when F is F_{MS} or F_{MM}. □

We next investigate the combined complexity of $\mathsf{RDC}(\mathcal{L}_Q, F)$ when F is F_{mono}.

THEOREM 7.2. *The combined complexity of* $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$ *is* $\#{\cdot}\mathsf{PSPACE}$*-complete under parsimonious reductions, when* $\mathcal{L}_Q$ *is CQ, UCQ,* $\exists$*FO*$^+$*or FO.* □

We verify the lower bound by parsimonious reduction from $\#\mathbf{QBF}$. The proof extends the counting argument given in the proof of Theorem 5.2. Given an instance $\varphi = \exists X \forall y_1 P_2 y_2 \ldots P_n y_n \psi(X, Y)$ of $\#\mathbf{QBF}$ over variables $X = \{x_1, \ldots, x_m\}$ and $Y = \{y_1, \ldots, y_n\}$, we define a distance function δ^{**} similar to its counterpart δ_{dis} given for QRD. More specifically, let $t = (u_1, \ldots, u_{m+n})$ and $s = (v_1, \ldots, v_{m+n})$ be any pair of $m+n$-arity tuples that encode two truth assignments of $X \cup Y$ variables, such that $t^{m+l} = s^{m+l}$ and $u_{m+l+1} \neq v_{m+l+1}$, where t^{m+l} and s^{m+l} are prefixes of t and s of length $m+l$, respectively. We want to use δ^{**} to ensure that $\delta^{**}(t, s) > 0$ if and only if formula

$P_{l+1}y_{l+1}\ldots P_ny_n\ \psi$ is satisfied by the truth assignment encoded by t^{m+l}, for variables $x_1,\ldots,x_m,y_1,\ldots,y_l$. We define $\delta^{**}_{\mathsf{dis}}$ by revising δ_{dis} as follows. (a) We let $\delta^{**}_{\mathsf{dis}}(t,s)=0$ if $t^m\neq s^m$, *i.e.,* t^m and s^m encode distinct truth assignments of X variables. Otherwise, let t^m be an arbitrary m-arity tuple that encodes a truth assignment of X, and $\breve{t}=(t^m,1,\ldots,1)$. For any tuple s with $s^m=t^m$, we define (b) $\delta^{**}_{\mathsf{dis}}(\breve{t},s)=(1/2)\cdot\delta_{\mathsf{dis}}(\breve{t},s)$ if $s=(t^m,1,v_2,\ldots,v_n)$; (c) $\delta^{**}_{\mathsf{dis}}(\breve{t},s)=4\cdot\delta_{\mathsf{dis}}(\breve{u},s)$ if $s=(t^m,0,v_2,\ldots,v_n)$; and moreover, (d) for any other pair of tuples t' and s', we let $\delta^{**}_{\mathsf{dis}}(t',s')=\delta_{\mathsf{dis}}(t',s')$.

It is easy to see that for each pair of t and s of $(m+n)-$arity tuples given above, $\delta_{\mathsf{dis}}(t,s)=1$ if and only if $\delta^{**}_{\mathsf{dis}}(t,s)>0$, for $l\in[0,n-1]$. Moreover, along the same line as Lemma 5.3, one can readily verify the following property of $\delta^{**}_{\mathsf{dis}}(t,s)$.

LEMMA 7.3. *Consider an instance $\varphi=\exists X\forall y_1P_2y_2\ldots P_ny_n\psi(X,Y)$ of #QBF, a truth assignment μ^m_X of X variables and a truth assignment μ^l_Y of variables in $\{y_1,\ldots,y_l\}$, for some $l\in[0,n-1]$. For any pair of tuples $t=(u_1,\ldots,u_{m+n})$ and $s=(v_1,\ldots,v_{m+n})$ that encode truth assignments of variables in φ, if $t^{m+l}=s^{m+l}=(\mu^m_X,\mu^l_Y)$ but $u_{m+l+1}\neq v_{m+l+1}$, then $P_{l+1}y_{l+1}\ldots P_ny_n\ \psi$ is true under μ^m_X and μ^l_Y if and only if $\delta^{**}_{\mathsf{dis}}(t,s)>0$, where t^{m+l} and s^{m+l} both encode μ^m_X and μ^l_Y.* □

Based on Lemma 7.3, we next prove Theorem 7.2.

PROOF. It suffices to show that $\mathsf{RDC}(\mathsf{CQ},F_{\mathsf{mono}})$ is #·PSPACE-hard and $\mathsf{RDC}(\mathsf{FO},F_{\mathsf{mono}})$ is in #·PSPACE. The lower bounds holds even when $\lambda=1$ and $k=1$.

(1) Lower bound. We verify the lower bound by parsimonious reduction from #QBF. Given an instance $\exists X\forall y_1P_2y_2\ldots P_ny_n\psi(X,Y)$ of #QBF, where $X=\{x_1,\ldots,x_m\}$ and $Y=\{y_1,\ldots,y_n\}$, we construct a database D, a CQ query Q, functions δ_{rel}, $\delta^{**}_{\mathsf{dis}}$ and F_{mono}, a positive integer k and a real number B, such that the number of truth assignments of X that satisfy φ equals the number of valid sets U for $(Q,D,k,F_{\mathsf{mono}},B)$. Intuitively, the reduction assures that for each truth assignment μ_X of X variables that satisfies φ, μ_X corresponds to a tuple $(t_X,1\ldots,1)$ such that $U=\{(t_X,1\ldots,1)\}$ is valid for $(Q,D,k,F_{\mathsf{mono}},B)$, where t_X is an m-arity tuple encoding μ_X; moreover, there exists no other valid set for $(Q,D,k,F_{\mathsf{mono}},B)$.

(1) The database D consists of a single relation $I_{01}=\{(0),(1)\}$ of Fig. 5, specified by schema $R_{01}(X)$ and encoding the Boolean domain.

(2) We define the CQ query Q as follows:

$$Q(\vec{x},\vec{y})=\bigwedge_{i\in[1,m]}R_{01}(x_i)\ \wedge\ \bigwedge_{j\in[1,n]}R_{01}(y_j).$$

Here $\vec{x}=(x_1,\ldots,x_m)$ and $\vec{y}=(y_1,\ldots,y_n)$. That is, Q generates all truth assignments of variables in $X\cup Y$. Obviously, $|Q(D)|=2^{m+n}$.

(3) We define relevance function δ_{rel} to be a constant function that returns 1 for any set U of tuples of R_Q, and use the distance function δ^{**} given above. We set $\lambda=1$, $k=1$ and $B=2^{n+1}/(2^{m+n}-1)$. Then for any set U consisting of k tuples of R_Q, we have that $F_{\mathsf{mono}}(U)=(1/(2^{m+n}-1))\cdot\sum_{t\in U,s\in Q(D)}\delta^{**}_{\mathsf{dis}}(t,s)$.

We next show that the number of truth assignments of X that satisfy φ is the same as the number of valid sets U for $(Q$, D, k, F_{mono}, $B)$. More specifically, we verify that if a truth assignment μ^m_X of X variables satisfies φ, then $\{(t^m,1\ldots,1)\}$ is a valid set for $(Q,D,k,F_{\mathsf{mono}},B)$, where t^m encodes μ^m_X, and moreover, there exists no other tuple s such that s^m encodes μ^m_X and $\{s\}$ is valid for $(Q,D,k,F_{\mathsf{mono}},B)$. For if it holds, then the encoding makes a parsimonious reduction from #QBF.

Assume first that the truth assignment μ_X^m of X variables satisfies φ. Let $\breve{t} = (\breve{t}^m, 1, \ldots, 1)$ such that $\breve{t}^m$ encodes μ_X^m. Then by Lemma 7.3, for each tuple $s = (\breve{t}^m, 0, v_2, \ldots, v_n)$, we have that $\delta^{**}_{\mathsf{dis}}(\breve{t}, s) = 4$. Note that there exist 2^{n-1} such tuples s. Then we have that $\sum_{s\in Q(D)} \delta^{**}_{\mathsf{dis}}(\breve{t}, s) \geq 4 \cdot 2^{n-1} = 2^{n+1}$. Thus $F_{\mathsf{mono}}(\{\breve{t}\}) \geq 2^{n+1}/(2^{m+n-1} - 1) \geq B$. Therefore, $\{\breve{t}\}$ is a valid set for $(Q, D, k, F_{\mathsf{mono}}, B)$. We next prove that for any other tuple t that is distinct from $\breve{t}$ such that $t^m = \breve{t}^m$ (*i.e.,* t^m encodes μ_X^m), we have that $F_{\mathsf{mono}}(\{t\}) < B$. Indeed, by the definition of $\delta^{**}_{\mathsf{dis}}$, for each tuple $t = (\breve{t}^m, u_{m+1}, \ldots, u_{m+n})$ such that $t \neq \breve{t}$, there are at most $2^n - 1$ tuples s such that $\delta^{**}_{\mathsf{dis}}(t, s) > 0$, and moreover, for each such tuple s, $s^m = t^m$. Consider the following two cases. For any tuple $t = (u_1, \ldots, u_{m+n})$ such that $t \neq \breve{t}$ and $t^m = \breve{t}^m$, (a) if $u_{m+1} = 0$, then by the definition of $\delta^{**}_{\mathsf{dis}}$, $\sum_{s\in Q(D)} \delta^{**}_{\mathsf{dis}}(t, s) = \sum_{s\in Q(D)\setminus\{\breve{t}\}} \delta^{**}_{\mathsf{dis}}(t, s) + \delta^{**}_{\mathsf{dis}}(t, \breve{t}) \leq 2^n - 2 + 4 = 2^n + 2 < 2^{n+1}$ (resp. $= 2^{n+1}$), when $n > 1$ (resp. when $n = 1$); and (b) otherwise $\sum_{s\in Q(D)} \delta^{**}_{\mathsf{dis}}(t, s) = \sum_{s\in Q(D)\setminus\{\breve{t}\}} \delta^{**}_{\mathsf{dis}}(t, s) + \delta^{**}_{\mathsf{dis}}(t, \breve{t}) \leq 2^n - 2 + 1/2 = 2^n - 3/2 < 2^{n+1}$. As a result, for each truth assignment μ_X^m that satisfies φ, there exists one and only one tuple $\breve{t} = (\breve{t}^m, 1, \ldots, 1)$ such that $\{\breve{t}\}$ is valid for $(Q, D, k, F_{\mathsf{mono}}, B)$, where $\breve{t}^m$ encodes μ_X^m.

(2.2) Upper bound. We show that $\mathsf{RDC}(\mathrm{FO}, F_{\mathsf{mono}})$ is in $\#\cdot$PSPACE. It suffices to prove that it is in PSPACE to verify whether a given set U is valid for $(Q, D, k, F_{\mathsf{mono}}, B)$. Indeed, consider the algorithm given in the proof of Theorem 5.2 for $\mathsf{QRD}(\mathrm{FO}, F_{\mathsf{mono}})$. Then its steps 4 and 5 can be used to check whether a given set U is valid. As shown there, steps 4 and 5 are in PSPACE. Therefore, $\mathsf{RDC}(\mathrm{FO}, F_{\mathsf{mono}})$ is in $\#\cdot$PSPACE. □

7.2. The Data Complexity of RDC

We next show that fixing queries reduces the complexity of RDC, to an extent:

(1) when F is F_{MS} or F_{MM}, the problem becomes #P-complete under parsimonious reductions, down from $\#\cdot$NP-complete (for CQ, UCQ and $\exists\mathrm{FO}^+$) and $\#\cdot$PSPACE-complete (for FO), as opposed to Theorem 7.1; and

(2) when F is F_{mono}, $\mathsf{RDC}(\mathcal{L}_Q, F)$ is #P-complete under polynomial Turing reductions, rather than $\#\cdot$PSPACE-complete, in contrast to Theorem 7.2.

Here #P is the class of functions that count the number of accepting paths of nondeterministic PTIME Turing machines, in the machine-based framework of [Valiant 1979]. It is known that #P = $\#\cdot$P [Durand et al. 2005], where $\#\cdot$P is the predicate-based counting class defined with a PTIME predicate (see Section 7.1).

Recall that a counting problem $\#A$ is *polynomial Turing reducible* to $\#B$ if there exists a PTIME function σ such that for all x, $|\{y \mid (x, y) \in A\}|$ is PTIME computable by making multiple calls to an oracle that computes $|\{z \mid (\sigma(x), z) \in B\}|$. We have so far used parsimonious reductions (Section 7.1), which are stronger than polynomial Turing reductions, *i.e.,* a parsimonious reduction from $\#A$ to $\#B$ is also a polynomial Turing reduction from $\#A$ to $\#B$, but not necessarily vice versa.

Note that a parsimonious reduction from $\#A$ to $\#B$ is also a PTIME reduction from its decision problem A to the decision problem B of $\#B$. Hence if the decision problem B of $\#B$ is in P, $\#B$ *cannot* be #P-complete under parsimonious reductions since for many NP-complete problems, *e.g.,* 3SAT, their counting problems are in #P. This is precisely the case for $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$ when $\mathcal{L}_Q$ is CQ, UCQ, $\exists\mathrm{FO}^+$ or FO (data complexity). Indeed, its decision problem $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{mono}})$ is in PTIME (Theorem 5.4). Thus $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$ is #P-complete under polynomial Turing reductions, but not under parsimonious reductions. In contrast, for F_{MS} or F_{MM}, $\mathsf{RDC}(\mathcal{L}_Q, F)$ is #P-complete under parsimonious reductions.

We first study the data complexity of $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{MS}})$ and $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{MM}})$

THEOREM 7.4. *the data complexity of* $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{MS}})$ *and* $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{MM}})$ *is* $\#\mathsf{P}$*-complete under parsimonious reductions for CQ, UCQ,* $\exists$*FO*$^+$*and FO.* □

The lower bound is verified by parsimonious reduction by $\#\mathsf{SAT}$: given an instance $\varphi(X) = C_1 \wedge \cdots \wedge C_l$ of 3SAT over variables $X = \{x_1, \ldots, x_m\}$, $\#\mathsf{SAT}$ is to count the number of truth assignments of X that satisfy φ. It is known that $\#\mathsf{SAT}$ is $\#\mathsf{P}$-complete (cf. [Papadimitriou 1994]). Given these, we prove Theorem 7.4 as follows.

PROOF. It suffices to show that $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{MS}})$ and $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{MM}})$ are $\#\mathsf{P}$-hard under parsimonious reductions, and that $\mathsf{RDC}(\mathrm{FO}, F_{\mathsf{MS}})$ and $\mathsf{RDC}(\mathrm{FO}, F_{\mathsf{MM}})$ are in $\#\mathsf{P}$.

(1) Lower bound. We show that $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{MS}})$ is $\#\mathsf{P}$-hard, even when $\lambda = 1$, by parsimonious reduction from $\#\mathsf{SAT}$. Given an instance $\varphi(X)$ of $\#\mathsf{SAT}$, we define the same D, Q, λ, δ_{rel}, δ_{dis} and F_{MS} as their counterparts given in the proof of Theorem 5.4 for $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MS}})$, and let $k = l$ and $B = l \cdot (l-1)$. Recall that in that proof, for each set $U \subseteq Q(D)$ such that $|U| = l$, $F_{\mathsf{MS}}(U) \geq l \cdot (l-1)$ if and only if tuples in U encode a truth assignment of X variables that satisfies φ. Thus the number of valid sets for (D, Q, k, F_{MS}, B) equals the number of truth assignments of X that satisfy φ.

We next show that $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{MM}})$ is $\#\mathsf{P}$-hard, also by parsimonious reduction from $\#\mathsf{SAT}$. Given an instance φ of $\#\mathsf{SAT}$, we construct the same $D, Q, \lambda, \delta_{\mathsf{rel}}, \delta_{\mathsf{dis}}$ and F_{MM} as their counterparts used in the proof of Theorem 5.4 for $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MM}})$, and let $k = l$ and $B = 1$. Recall that in that proof, for each set $U \subseteq Q(D)$ such that $|U| = l$, $F_{\mathsf{MM}}(U) \geq 1$ if and only if the tuples in U encode a truth assignment of X variables that satisfies φ. Thus the number of valid sets for $(D, Q, k, F_{\mathsf{MM}}, B)$ is equal to the number of truth assignments of X variables that satisfy φ.

(2) Upper bound. We show that $\mathsf{RDC}(\mathrm{FO}, F)$ is in $\#\mathsf{P}$ for max-sum and max-min diversification. To do this, we only need to show that it is in PTIME to verify whether a given set U is valid for (Q, D, k, F, B), when $F = F_{\mathsf{MS}}$ or $F = F_{\mathsf{MM}}$, by the definition of $\#\cdot\mathsf{P}$ (recall that $\#\cdot\mathsf{P} = \#\mathsf{P}$). Indeed, given a set U, it is in PTIME to check whether $U \subseteq Q(D)$ for a fixed query Q in FO, and is in PTIME to check whether $|U| = k$. Moreover, checking whether $F(U) \geq B$ is also in PTIME when F is F_{MS} or F_{MM}. Thus $\mathsf{RDC}(\mathrm{FO}, F)$ is in $\#\mathsf{P}$. □

We next investigate the data complexity of $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$.

THEOREM 7.5. *The data complexity of* $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$ *is* $\#\mathsf{P}$*-complete under Turing reductions for CQ, UCQ,* $\exists$*FO*$^+$*and FO.* □

To show the lower bound, we reduce from the following problems, and use a lemma.

(1) $\#\mathsf{SSP}$ (the #subset sum problem): Given a finite set W, a function $\pi : W \to \mathbb{N}$ and a natural number $d \in \mathbb{N}$, it is to count the number of subsets $T \subseteq W$ such that $\sum_{w \in T} \pi(w) = d$. It is known that $\#\mathsf{SSP}$ is $\#\mathsf{P}$-complete under parsimonious reduction [Berbeglia and Hahn 2010].

(2) $\#\mathsf{SSPk}$: Given a finite set W, a function $\pi : W \to \mathbb{N}$ and natural numbers $d, l \in \mathbb{N}$, $\#\mathsf{SSPk}$ is to count the number of subsets $T \subseteq W$ such that $|T| = l$ and $\sum_{w \in T} \pi(w) = d$.

We first verify that $\#\mathsf{SSPk}$ is $\#\mathsf{P}$-hard by parsimonious reduction from $\#\mathsf{SSP}$ and then show that $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{mono}})$ is $\#\mathsf{P}$-hard by Turing reduction from $\#\mathsf{SSPk}$.

LEMMA 7.6. *The* $\#\mathsf{SSPk}$ *problem is* $\#\mathsf{P}$*-complete under parsimonious reductions.* □

PROOF. To see that #SSPk is in #P, observe that given a finite set W, a function $\pi : W \to \mathbb{N}$, a subset $T \subseteq W$, $d \in \mathbb{N}$ and $l \in \mathbb{N}$, it is in PTIME to check whether $|T| = l$ and $\sum_{w \in W} \pi(w) = d$. Thus #SSPk is in #P by the definition of #·P (recall #·P = #P).

We next show that #SSPk is #P-hard by parsimonious reduction from #SSP. Given an instance W, π and d of #SSP, we construct W', π', d' and l such that the number of subsets $T \subseteq W$ with $\sum_{w \in T} \pi(w) = d$ equals the number of $T' \subseteq W'$ with $|T'| = l$ and $\sum_{w' \in T'} \pi'(w') = d'$. Let $W = \{w_1, \ldots, w_n\}$ (hence $|W| = n$). Denote by m the number of decimal digits in $\sum_{w \in W} \pi(w)$. Then for each subset T of W, $\sum_{w \in T} \pi(w)$ can be also represented as an m-digit integer. We next give the reduction.

(1) We define $W' = \{(w_i, 1), (w_i, 0) \mid i \in [1, n]\}$. That is, we include two elements $(w_i, 1)$ and $(w_i, 0)$ in W' for each $w_i \in W$, where $i \in [1, n]$.

(2) We define π' as a function from W' to $\mathbb{N}$. Intuitively, for each $w' \in W'$, we define $\pi'(w')$ to be an $n + m$-digit integer, where the first n digits in $\pi'(w')$ encode w_i for $i \in [1, n]$, where $w' = (w_i, 1)$ or $w' = (w_i, 0)$; moreover, the last m-digit integer in $\pi'(w')$ either equals $\pi(w_i)$ for some w_i if $w' = (w_i, 1)$, or equals 0 when $w = (w_i, 0)$. More specifically, for each $w' \in W'$, we define $\pi'(w')$ to be an $n + m$-digit integer $u_1 \ldots u_n v_1 \ldots v_m$, such that (a) if $w' = (w_i, 1)$ for some $w_i \in W$, then $u_i = 1$, and for each $j \in [1, n]$, if $j \neq i$, then $u_j = 0$; and moreover, the m-digit integer $v_1 \ldots v_m$ equals $\pi(w_i)$; and (b) if $w' = (w_j, 0)$ for some $w_j \in W$, then $u_j = 1$, and for each $i \in [1, n]$, if $i \neq j$, then $u_i = 0$, and furthermore, for all $i \in [1, m]$, $v_i = 0$.

(3) We define d' to be an $n + m$-digit integer $u_1^* \ldots u_n^* v_1^* \ldots v_m^*$, where for each $i \in [1, n]$, $u_i^* = 1$, and moreover, the m-digit integer $v_1^* \ldots v_m^*$ equals d. Indeed, d is an m-digit integer since $d \leq \sum_{w \in W} \pi(w)$. Finally, we define $l = n$, *i.e.,* $l = |W|$.

We next show that this is indeed a parsimonious reduction, *i.e.,* the number of subsets $T \subseteq W$ such that $\sum_{w \in T} \pi(w) = d$ is equal to the number of subsets $T' \subseteq W'$ with $|T'| = l$ and $\sum_{w' \in T'} \pi(w') = d'$. Assume that there exists a set $T' \subseteq W'$ such that $|T'| = l = n$ and $\sum_{w' \in T'} \pi'(w') = d'$. Then by the definitions of d' and π', the sum of last m-digit integers in all $\pi'(w')$ for $w' \in T'$ is equal to d, and moreover, $d = \sum_{(w_i, 1) \in T'} \pi(w_i)$. Let T be the set consisting of all w_i such that $(w_i, 1) \in T'$. Then $\sum_{w_i \in T} \pi(w_i) = \sum_{(w_i, 1) \in T'} \pi(w_i) = d$. Conversely, assume that there exists a subset $T \subseteq W$ such that $\sum_{w \in T} \pi(w) = d$. Define $T' = \{(w_i, 1) \mid w_i \in T\} \cup \{(w_j, 0) \mid w_j \in W \setminus T\}$. Obviously, $|T'| = l = n$. Again by the definitions of π' and d', one can verify that $\sum_{w' \in T'} \pi'(w') = \sum_{(w_i, 1) \in T'} \pi(w_i) = d'$. Obviously, the reduction is parsimonious.

Putting these together, we have that #SSPk is #P-complete. □

Using Lemma 7.6, we next prove Theorem 7.5.

PROOF. It suffices to show that RDC(CQ, F_{mono}) is #P-hard under polynomial Turing reductions, and that RDC(FO, F_{mono}) is in #P, even when $\lambda = 0$.

(1) Lower bounds. We show that RDC(CQ, F_{mono}) is #P-hard by polynomial Turing reduction from #SSPk, which has been shown #P-complete by Lemma 7.6. Denote by $\mathsf{COUNT}_{\mathsf{RDC}}(Q, D, k, F_{\mathsf{mono}}, B)$ the oracle that given $Q, D, k, F_{\mathsf{mono}}$ and B, returns the number of valid sets U for $(Q, D, k, F_{\mathsf{mono}}, B)$. To show that #SSPk is polynomial time Turing reducible to RDC(CQ, F_{mono}), it suffices to show that there exists a PTIME algorithm for computing #SSPk by calling $\mathsf{COUNT}_{\mathsf{RDC}}(Q, D, k, F_{\mathsf{mono}}, B)$ a polynomial number of times, by the notion of polynomial Turing reductions given earlier.

Observe the following. Given a finite set W, a function $\pi : W \to \mathbb{N}$ and natural numbers d and l, let X be the number of subsets T of W such that $\sum_{w \in T} \pi(w) = d$ and $|T| = l$, Y be the number of subsets T' of W such that $\sum_{w \in T'} \pi(w) \geq d$ and $|T'| = l$, and

Z be the number of subsets T'' of W such that $\sum_{w \in T''} \pi(w) \geq d+1$ and $|T''| = l$. Then $X = Y - Z$. Based on this, we construct the polynomial Turing reduction from #SSPk to RDC(CQ, F_{mono}) as follows. Given an instance W, π, l and d of #SSPk, we first construct Q, D, δ_{rel}, δ_{dis}, F_{mono}, k and B in PTIME, such that the number of subsets T of W with $|T| = l$ and $\sum_{w \in T} \pi(w) \geq d$ equals the number of valid sets U for $(Q, D, k, F_{\mathsf{mono}}, B)$. As discussed above, we can find the solution for #SSPk, *i.e.,* the number of subsets T of W with $|T| = l$ and $\sum_{w \in T} \pi(w) = d$, by calling the oracle $\mathsf{COUNT}_{\mathsf{RDC}}$ *twice*, for computing the numbers X and Y of valid sets for $(Q, D, k, F_{\mathsf{mono}}, B)$ and $(Q, D, k, F_{\mathsf{mono}}, B+1)$, respectively.

We next give the transformation from #SSPk to RDC(CQ, F_{mono}).

(1) The database D consists of a single relation $I_W = \{(w) \mid w \in W\}$ of schema $R_W(W)$.

(2) We define query Q as the identity query on R_W instances.

(3) We define δ_{rel} as follows. For each tuple $t = (w) \in Q(D)$, we let $\delta_{\mathsf{rel}}(t, Q) = \pi(w)$. Furthermore, for any other tuple t' of R_Q, we define $\delta_{\mathsf{rel}}(t', Q) = 0$. Moreover, we take δ_{dis} as a constant function that returns 0 for each pair of tuples of R_Q. We set $\lambda = 0$, and hence for each set U of tuples of R_Q, $F_{\mathsf{mono}}(U) = \sum_{t \in U} \delta_{\mathsf{rel}}(t, Q)$.

(4) Finally, we set $k = l$ and $B = d$.

We next show that the number of subsets T of W such that $|T| = l$ and $\sum_{w \in T} \pi(w) \geq d$ equals the number of valid sets U for $(Q, D, k, F_{\mathsf{mono}}, B)$. Note that $k = l$ and $B = d$. Then by the definition of δ_{rel}, for each set $U \subseteq Q(D)$ such that $|U| = k$, $F_{\mathsf{mono}}(U) = \sum_{(w) \in U} \pi(w) \geq B$ if and only if for the set $T = \{w \mid (w) \in U\}$, we have that $|T| = l$ and $\sum_{w \in T} \pi(w) \geq d$. From this we have a PTIME algorithm for computing #SSPk by calling $\mathsf{COUNT}_{\mathsf{RDC}}(Q, D, k, F_{\mathsf{mono}}, B)$ (denoted by X) and $\mathsf{COUNT}_{\mathsf{RDC}}(Q, D, k, F_{\mathsf{mono}}, B+1)$ (denoted by Y). Then $X - Y$ is the solution for #SSPk.

(2) Upper bound. We verify that RDC(FO, F_{mono}) is in #P, by showing that it is in PTIME to verify whether a given set U is valid for $(Q, D, k, F_{\mathsf{mono}}, B)$. Indeed, $Q(D)$ and F_{mono} are both PTIME computable since Q is fixed. Thus it is in PTIME to check whether $U \subseteq Q(D)$, $|U| = k$ and $F_{\mathsf{mono}}(U) \geq B$. Hence RDC(FO, F_{mono}) is in #P. □

Summary. Taking the results of Sections 5, 6 and 7 together, we can find the following.

(1) Both query languages and objective functions have impact on the combined complexity of query result diversification. More specifically, (a) when F is F_{MS} or F_{MM}, the diversification problems for FO have a higher combined complexity than their counterparts for CQ, UCQ and $\exists$FO$^+$; and (b) when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$, F_{mono} makes the diversification problems harder than F_{MS} and F_{MM}.

(2) When the objective is given by mono-objective formulation, the objective function dominates the combined complexity. Indeed, the combined complexity bounds of these problems are independent of whether we take CQ, UCQ, $\exists$FO$^+$or FO as $\mathcal{L}_Q$.

(3) When it comes to data complexity, query languages make no difference, while objective functions determine the complexity. Indeed, the data complexity bounds of these problems remain unchanged when $\mathcal{L}_Q$ is CQ, UCQ, $\exists$FO$^+$or FO. Moreover, when the objective is given by F_{mono}, QRD and DRP become tractable, but it is not the case when the objective is for F_{MS} or F_{MM}. That is, the data complexity is *inherent to* result diversification itself, rather than a consequence of the complexity of query languages.

8. SPECIAL CASES OF QUERY RESULT DIVERSIFICATION

In this section we identify and investigate several special cases of QRD, DRP and RDC. The reason for studying these is twofold. (1) The results of Sections 5, 6 and 7 tell us that these problems have rather high complexity. This suggests that we find their special yet practical cases that are tractable. (2) We want to further understand the impact of various parameters of these problems on their complexity, including query languages with low complexity, relevance functions δ_{rel}, distance functions δ_{dis}, and the bound k for selecting query answers.

Due to the space constraint, we refer the interested reader to the electronic appendix for the proofs of the results to be presented in this section and Section 9.

Identity queries. We first consider the case when $\mathcal{L}_Q$ consists of identity queries only, *i.e.,* when query Q is of the following form:

$$Q(\bar{x}) = R(\bar{x}),$$

where R is a relation atom, and $|\bar{x}|$ is the arity of R. Note that for any instance D of schema R, $D = Q(D)$. As remarked early, in this setting QRD was shown to be NP-hard by [Gollapudi and Sharma 2009] when the objective is for max-sum diversification and max-min diversification. No previous work has studied QRD for mono-objective formulation, or DRP and RDC for any of the three objective functions.

We show that identity queries reduce the complexity of these problems to an extent.

(1) When the objective is given by mono-objective formulation, QRD and DRP are tractable, as opposed to the NP-hardness of QRD for F_{MS} and F_{MM} [Gollapudi and Sharma 2009], and RDC becomes #P-complete. In contrast, these problems are PSPACE-complete, PSPACE-complete, and #·PSPACE-complete (Theorems 5.2, 6.2 and 7.2), respectively, when $\mathcal{L}_Q$ is CQ. This further verifies that query languages have impact on the complexity of diversification.

(2) In contrast, when the objective is for max-sum or max-min diversification, the combined complexity and data complexity of these problems are the same as their counterparts when $\mathcal{L}_Q$ is CQ. In other words, in this setting, query languages with a low complexity (for its membership problem) do not simplify the analyses of diversification.

COROLLARY 8.1. *For identity queries, the combined complexity and data complexity of* QRD, DRP *and* RDC *coincide. More specifically,*

— QRD$(\mathcal{L}_Q, F_{\mathsf{MS}})$ *and* QRD$(\mathcal{L}_Q, F_{\mathsf{MM}})$ *are* NP*-complete,*
— DRP$(\mathcal{L}_Q, F_{\mathsf{MS}})$ *and* DRP$(\mathcal{L}_Q, F_{\mathsf{MM}})$ *are* coNP*-complete, and*
— RDC$(\mathcal{L}_Q, F_{\mathsf{MS}})$ *and* RDC$(\mathcal{L}_Q, F_{\mathsf{MM}})$ *are* #P*-complete under parsimonious reductions,*

for both combined complexity and data complexity, while

— QRD$(\mathcal{L}_Q, F_{\mathsf{mono}})$ *is in* PTIME*,*
— DRP$(\mathcal{L}_Q, F_{\mathsf{mono}})$ *is in* PTIME*, and*
— RDC$(\mathcal{L}_Q, F_{\mathsf{mono}})$ *is* #P*-complete under polynomial Turing reductions,*

for both combined complexity and data complexity, which are the same as their data complexity given in Theorems 5.4, 6.4 and 7.5, respectively. □

When λ = 0. We next focus on the impact of the relevance and diversity requirements on the complexity of query result diversification analyses. We first consider the case when $\lambda = 0$, *i.e.,* the objective function F is defined in terms of the relevance function δ_{rel} only. We find that the diversity requirement has higher impact on the complexity than relevance. Indeed, dropping distance functions δ_{dis} simplifies the analyses of these problems to an extent. This is consistent with the observation of [Vieira et al. 2011].

(1) When the objective function is F_{MS} or F_{MM}, QRD and DRP become tractable for

fixed Q. Moreover, RDC is in FP for F_{MM}, where FP is the class of all functions that can be computed in PTIME (cf. [Papadimitriou 1994]). That is, these problems have lower data complexity.

(2) When the objective is given by F_{mono}, the combined complexity analyses of these problems become simpler, when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$.

THEOREM 8.2. *When λ = 0, For F_{MS} and F_{MM}, the combined complexity bounds of* QRD*,* DRP *and* RDC *remain the same as their counterparts given in Theorems 5.1, 6.1 and 7.1, respectively. In contrast, when $\mathcal{L}_Q$ is CQ, UCQ, $\exists$FO$^+$or FO, the data complexity bounds of these problems are*

— *in* PTIME *for* QRD$(\mathcal{L}_Q, F_{\mathsf{MS}})$ *and* QRD$(\mathcal{L}_Q, F_{\mathsf{MM}})$*,*
— *in* PTIME *for* DRP$(\mathcal{L}_Q, F_{\mathsf{MS}})$ *and* DRP$(\mathcal{L}_Q, F_{\mathsf{MM}})$*, and*
— #P*-complete for* RDC$(\mathcal{L}_Q, F_{\mathsf{MS}})$ *under polynomial Turing reductions, but in* FP *for* RDC$(\mathcal{L}_Q, F_{\mathsf{MM}})$*.*

For F_{mono}, the combined complexity becomes

— NP*-complete for* QRD$(\mathcal{L}_Q, F_{\mathsf{mono}})$ *when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$, and* PSPACE*-complete when $\mathcal{L}_Q$ is FO;*
— coNP*-complete for* DRP$(\mathcal{L}_Q, F_{\mathsf{mono}})$ *when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$, and* PSPACE*-complete for FO; and*
— $\#\cdot$NP*-complete for* RDC$(\mathcal{L}_Q, F_{\mathsf{mono}})$ *when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$, and* $\#\cdot$PSPACE*-complete for FO.*

The data complexity bounds of these problems remain the same as their counterparts given in Theorems 5.4, 6.4, 7.4 and 7.5, respectively, when $\mathcal{L}_Q$ is CQ, UCQ, $\exists$FO$^+$or FO. □

When λ = 1. In contrast to Theorem 8.2, we show below that dropping the relevance function δ_{rel} does not simplify the analyses. Indeed, when the objective function is defined with only the diversity function δ_{dis}, both the combined complexity and data complexity of QRD, DRP and RDC remain the same as their counterparts when both relevance and diversity are taken into account. This further verifies that the diversity requirement δ_{dis} dominates the complexity of these problems. These results, however, need new proofs and are not corollaries of the previous results.

THEOREM 8.3. *When λ = 1, the combined complexity of Theorems 5.1, 5.2, 6.1, 6.2, 7.1 and 7.2 and the data complexity of Theorems 5.4, 6.4, 7.4 and 7.5 remain unchanged for* QRD*,* DRP *and* RDC*, respectively.* □

When k is a predefined constant. Finally, we study the impact of the cardinality $|U|$ of selected sets U of query answers on the analyses of query result diversification. When $|U|$ is fixed to be a predefined *constant* k, the result below tells us the following.

(1) When Q is also fixed, QRD, DRP and RDC are all tractable. That is, fixing the size of U simplifies their data complexity analyses.

(2) In contrast, fixing k does not simplify the combined complexity analyses of these problems. Indeed, all the combined complexity bounds of these problems remain the same as their counterparts when k is not required to be a constant.

COROLLARY 8.4. *For a predefined constant k,*

— *the combined complexity bounds given in Theorems 5.1, 5.2, 6.1, 6.2, 7.1 and 7.2 are unchanged for* QRD*,* DRP *and* RDC*, respectively; and*
— *the data complexity is in*
  — PTIME *for* QRD*,*

— PTIME *for* DRP*, and*
— FP *for* RDC*,*

no matter whether for F_{MS}*,* F_{MM} *or* F_{mono}*, and for CQ, UCQ,* $\exists$*FO*$^+$*or FO.*

Summary. From the results that we have got so far, we can see the following.

(1) The impact of $\mathcal{L}_Q$. When the objective function is given by F_{MS} or F_{MM}, QRD, DRP and RDC have higher combined complexity for FO than for CQ, UCQ and $\exists$FO$^+$(Theorems 5.1, 6.1 and 7.1). In contrast, the complexity bounds remain intact for CQ or the class of identity queries (Corollary 8.1). When considering F_{mono}, the combined complexity bounds of these problems remain unchanged for all query languages CQ, UCQ, $\exists$FO$^+$and FO (Theorems 5.2, 6.2 and 7.2). In contrast, when for the class of identity queries, these problems become simpler (Corollary 8.1). Note that the query languages have no impact on the data complexity of these problems (Theorems 5.4, 6.4, 7.4, 7.5 and Corollary 8.1).

(2) The impact of δ_{rel} and δ_{dis}. The complexity of diversification also arises from the diversity requirement. The absence of the distance function δ_{dis} simplifies (a) the data complexity analyses of QRD and DRP for F_{MS} or F_{MM}, (b) the data complexity of RDC for F_{MM}, and (c) the combined complexity analyses of all these problems for F_{mono} (Theorem 8.2). In contrast, the absence of δ_{rel} has no impact on the complexity (Theorem 8.3).

(3) The impact of k. When k is a fixed constant, the data complexity analyses of QRD, DRP and RDC become tractable, no matter whether objective function is given by F_{MS}, F_{MM} or F_{mono} (Corollary 8.4).

9. INCORPORATING COMPATIBILITY CONSTRAINTS

In this section, we study the impact of compatibility constraints on the analyses of query result diversification. We first introduce a class of compatibility constraints, and extend the diversification model of Section 3 by incorporating these constraints. In the presence of such constraints, we then re-investigate QRD, DRP and RDC in all the settings of the previous sections (Sections 5, 6, 7 and 8).

Compatibility constraints. We first define a class of compatibility constraints. Consider a database D, a query Q in a language $\mathcal{L}_Q$, and a predefined constant $m \geq 2$. We define a class $\mathcal{C}_m$ of *compatibility constraints* on subsets $U \subseteq Q(D)$. Let R_Q denote the schema of query results $Q(D)$. A constraint φ in $\mathcal{C}_m$ is of the form:

$$\forall t_1, \ldots, t_l : R_Q \ \big(\chi(t_1, \ldots, t_l) \rightarrow \exists s_1, \ldots, s_h : R_Q \ \xi(t_1, \ldots, t_l, s_1, \ldots, s_h)\big).$$

Here (1) l and h are in the range $[0, m]$, (2) t_i and s_j are tuple variables denoting a tuple of R_Q, and (3) χ and ξ are conjunctions of predicates of the form (a) $\rho[A] = \varrho[B]$ or $\rho[A] \neq \varrho[B]$, or (b) $\rho[A] = c$ or $\rho[A] \neq c$, where A and B are attributes in R_Q, ρ and ϱ range over tuples t_i and s_j for $i \in [0, l]$ and $j \in [0, h]$, and c is a constant.

We say that a set $U \subseteq Q(D)$ *satisfies* φ, denoted by $U \models \varphi$, if for all tuples $t_1, \ldots, t_l$ in U that satisfy the predicates in χ following the standard semantics of first-order logic, there must exist tuples $s_1, \ldots s_h$ in U such that all the predicates in ξ are also satisfied. We say that D *satisfies* a set Σ of constraints in $\mathcal{C}_m$ if for each $\varphi \in \Sigma$, $U \models \varphi$.

Class $\mathcal{C}_m$ suffices to express compatibility constraints commonly found in practice, to specify what items should be picked together when we select top-k tuples, and what items have conflict with each other, as illustrated by the following example.

Example 9.1. Consider a query Q_1 posed on database D_1 to find items for shopping. The selected items are specified by a relation schema R_{Q1} with attributes item, price, etc (see Example 1.1). The compatibility constraint ρ_1 below is defined on subsets of results $Q_1(D_1)$. It assures that if one buys items a and b, then she also needs to buy c. That is, in the set U of top-k items recommended, if a and b are in U, then so is c.

$$\rho_1 = \forall\, t_1, t_2 : R_{Q1}\ \big((t_1[\mathsf{item}] = a \wedge t_2[\mathsf{item}] = b) \rightarrow \exists\, s : R_{Q1}(s[\mathsf{item}] = c)\big).$$

As another example, consider a query Q_2 posed on a database D_2 for course selection [Koutrika et al. 2009; Parameswaran et al. 2010]. The schema of query result $Q_2(D_2)$ is denoted by R_{Q2}, including attributes id and title, among other things. The compatibility constraint ρ_2 below is defined on instances of R_{Q2}, *i.e.,* sets $U \subseteq Q_2(D_2)$. It asserts that if course CS450 is taken, then so must be its prerequisites CS220 and CS350.

$$\rho_2 = \forall\, t_1 : R_{Q2}\ \big(t_1[\mathsf{id}] = \text{CS450} \rightarrow \exists\, s_1, s_2 : R_{Q2}(s_1[\mathsf{id}] = \text{CS220} \wedge s_2[\mathsf{id}] = \text{CS350})\big).$$

Now consider a query Q_3 posed on a database D_3 for basketball team formation [Lappas et al. 2009]. The schema of $Q_3(D_3)$ is R_{Q3}, including attributes id, position, etc. We use the following constraint ρ_3 to assure that at most two centers are needed for the team, *i.e.,* no more than three centers may be included in any top-k sets $U \subseteq Q_3(D_3)$.

$$\begin{aligned}\rho_3 \ =\ \forall\, t_1, t_2, t_3 : R_{Q3}\ \big(& t_1[\mathsf{position}] = \mathsf{center} \wedge t_2[\mathsf{position}] = \mathsf{center} \wedge \\ & t_1[\mathsf{id}] \neq t_2[\mathsf{id}] \wedge t_1[\mathsf{id}] \neq t_3[\mathsf{id}] \wedge t_2[\mathsf{id}] \neq t_3[\mathsf{id}] \rightarrow t_2[\mathsf{position}] \neq \mathsf{center})\big).\end{aligned}$$

Observe that compatibility constraints φ_1, φ_2 and φ_3 may not be expressible in the query languages $\mathcal{L}_Q$ for Q_1, Q_2 and Q_3, when, *e.g.,* $\mathcal{L}_Q$ is CQ. □

One can see that constraints of $\mathcal{C}_m$ have a form similar to tuple generating dependencies (TGDs) that have been well studied for databases (see, *e.g.,* [Abiteboul et al. 1995] for TGDs), except that the number of tuples in each constraint of $\mathcal{C}_m$ is bounded by a predefined constant m, such as our familiar functional dependencies and inclusion dependencies, which are bounded by constant 2 and can be expressed in $\mathcal{C}_m$ when $m \geq 2$. Moreover, one can readily verify that constraints of $\mathcal{C}_m$ are in PTIME, *i.e.,* for any set $\Sigma \subseteq \mathcal{C}_m$ and any set $U \in Q(D)$, it takes at most PTIME in $|U|$ and $|\Sigma|$ to determine whether $U \models \Sigma$, because the number of tuples in each $\varphi \in \Sigma$ is bounded by m.

Query result diversification revisited. We are now ready to revise query result diversification by incorporating constraints of $\mathcal{C}_m$. Given a query Q in a query language $\mathcal{L}_Q$, a database D, a positive integer k, an objective function F, and in addition, a set Σ of compatibility constraints in $\mathcal{C}_m$ defined on subsets of $Q(D)$, *query result diversification in the presence of compatibility constraints* aims to find a set $U \subseteq Q(D)$ such that (a) $|U| = k$, (b) $F(U)$ is maximum, and moreover, (c) $U \models \Sigma$. Compared to diversification in the absence of compatibility constraints (see Section 3), the top-k set U selected is additionally required to satisfy all the constraints in Σ.

In the presence of Σ, we revise the following notions introduced in Section 4.

(1) Given Q, D, k and Σ, we say that a set $U \subseteq Q(D)$ is a *candidate set for* (Q, D, Σ, k) if $|U| = k$ and $U \models \Sigma$.

(2) Given a real number B and an objective function F, we say that a set $U \subseteq Q(D)$ is *valid for* (Q, D, Σ, k, F, B) if $|U| = k$, $U \models \Sigma$ and $F(U) \geq B$.

(3) We say that $\mathsf{rank}(U) = r$ for a positive integer r if there exists a collection $\mathcal{S}$ of $r - 1$ distinct candidate sets for (Q, D, Σ, k) such that (a) for all $S \in \mathcal{S}$, $F(S) > F(U)$; and (b) for any candidate set S' for (Q, D, Σ, k), if $S' \notin \mathcal{S}$, then $F(U) \geq F(S')$.

Based on these, we revise the statements of problems QRD, DRP and RDC as follows.

(1) Problem $\mathsf{QRD}(\mathcal{L}_Q, F)$ is to decide, given D, $Q \in \mathcal{L}_Q$, F, B and in addition, a set Σ of compatibility constraints in $\mathcal{C}_m$, whether there exists a valid set for (Q, D, Σ, k, F, B).

(2) Problem $\mathsf{DRP}(\mathcal{L}_Q, F)$ is to decide, given D, $Q \in \mathcal{L}_Q$, F, B, Σ, and a candidate set U for (Q, D, Σ, k), whether $\mathsf{rank}(U) \leq r$, where r is a positive integer constant.

(3) Problem $\mathsf{RDC}(\mathcal{L}_Q, F)$ is to count the number of valid sets for (Q, D, Σ, k, F, B).

We next investigate these problems in the presence of compatibility constraints. We first establish their combined complexity, and then provide their data complexity. Finally, we study these problems in the special cases identified in Section 8.

Combined complexity. The good news is that compatibility constraints do not complicate the combined complexity analyses of the result diversification problems. Indeed, constraints of $\mathcal{C}_m$ can be validated in PTIME, and hence all the upper bounds given in Sections 5, 6 and 7 for combined complexity remain intact.

COROLLARY 9.2. *In the presence of compatibility constraints of $\mathcal{C}_m$, the combined complexity bounds of Theorems 5.1, 5.2, 6.1, 6.2, 7.1 and 7.2 remain unchanged for* QRD*,* DRP *and* RDC*.* □

Data complexity. In the presence of compatibility constraints, we study the data complexity of these problems, *i.e.,* when query Q and compatibility constraints Σ are predefined and *fixed*, while database D may vary. We show that the presence of Σ makes the problems harder, to an extent.

(1) When the objective is given by mono-object formulation, QRD and DRP become NP-complete and coNP-complete, respectively, as opposed to their tractability in the absence of compatibility constraints (Theorem 5.4 and 6.4), and DRP becomes #P-complete under parsimonious reductions, rather than under polynomial Turing reductions (Theorem 7.5). That is, although compatibility constraints of $\mathcal{C}_m$ can be validated in PTIME, they impose additional requirements on the selection of top-k sets based on F_{mono} and hence, complicate the analyses of these problems when query Q is fixed. These data complexity results hold no matter whether for CQ, UCQ, $\exists$FO$^+$and FO.

(2) In contrast, when the objective is for max-sum or max-min diversification, the data complexity results of these problems remain the same as their counterparts in the absence of compatibility constraints.

THEOREM 9.3. *In the presence of compatibility constraints of $\mathcal{C}_m$, the data complexity bounds of Theorems 5.4, 6.4 and 7.4 remain unchanged for* QRD*,* DRP *and* RDC*, respectively, for F_{MS} and F_{MM}. However, for F_{mono},*

— QRD *becomes* NP*-complete;*
— DRP *becomes* coNP*-complete; and*
— RDC *becomes* #P*-complete under parsimonious reductions.* □

Special cases. We next investigate the special cases of Section 8 in the presence of compatibility constraints of $\mathcal{C}_m$. We show that compatibility constraints make the analyses of query result diversification more complicated: all the tractable cases we have seen in Section 8 except one (when the bound k is a constant) become intractable, although the compatibility constraints of $\mathcal{C}_m$ are simple enough to be validated in PTIME.

Identity queries. We first consider the case when $\mathcal{L}_Q$ consists of identity queries (see Section 8 for the details of identity queries). In this setting, compatibility constraints complicate the combined and data complexity analyses of query result diversification when F is F_{mono}. Indeed, $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{mono}})$, $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{mono}})$ and $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{mono}})$ become NP-complete, coNP-complete and #P-complete under parsimonious reductions, respectively, for both their combined complexity and data complexity, as opposed to PTIME, PTIME and #P-complete under polynomial Turing reductions, respectively, in the absence of such constraints (Corollary 8.1). This is because for an identity query Q and a database D, while $Q(D) = D$ and $F_{\mathsf{mono}}(U)$ for $U \subseteq Q(D)$ is computable in PTIME, the additional requirements imposed by compatibility constraints make checking and counting valid sets more intricate, similar to the complication introduced by the constraints to the data complexity analyses of these problems (Theorem 9.3). In contrast, when F is F_{MS} or F_{MM}, even the data complexity analyses of these problems are already intractable in the absence of compatibility constraints, and the compatibility constraints do not increase their complexity bounds.

COROLLARY 9.4. *For identity queries, in the presence of compatibility constraints of $\mathcal{C}_m$, both the combined complexity and data complexity of Corollary 8.1 remain unchanged for* QRD, DRP, *and* RDC *for* F_{MS} *and* F_{MM}.

However, when it comes to F_{mono},

— QRD *becomes* NP-*complete;*
— DRP *becomes* coNP-*complete; and*
— RDC *becomes* #P-*complete under parsimonious reductions,*

for both combined and data complexity. □

When $\lambda = 0$. We next study the impact of the compatibility constraints on the complexity of query result diversification analyses when $\lambda = 0$, *i.e.,* when the objective function F is defined in terms of the relevance function δ_{rel} only. The results below tell us the following. The presence of compatibility constraints has no impact on the combined complexity of $\mathsf{QRD}(\mathcal{L}_Q, F)$, $\mathsf{DRP}(\mathcal{L}_Q, F)$ and $\mathsf{DRP}(\mathcal{L}_Q, F)$, but the constraints do make the data complexity analyses harder.

COROLLARY 9.5. *For $\lambda = 0$, in the presence of compatibility constraints of $\mathcal{C}_m$, the combined complexity bounds given in Theorem 8.2 remain unchanged for* QRD, DRP *and* RDC, *while the data complexity becomes*

— NP-*complete for* QRD*;*
— coNP-*complete for* DRP*; and*
— #P-*complete for* RDC *under parsimonious reductions,*

no matter for F_{MS}, F_{MM} *and* F_{mono}, *and for CQ, UCQ, $\exists$FO$^+$and FO.* □

When $\lambda = 1$. Similarly, when F is defined in terms of distance function δ_{dis} only, compatibility constraints complicate the data analyses of QRD, DRP and RDC for F_{mono}. For F_{MS} and F_{MM}, these problems are already NP-complete, coNP-complete and #P-complete under parsimonious reductions, respectively, in the absence of compatibility constraints (Theorem 8.3), and these data complexity bounds remain intact in the presence of the constraints.

COROLLARY 9.6. *For $\lambda = 1$, in the presence of compatibility constraints of $\mathcal{C}_m$, the combined complexity bounds given in Theorem 8.3 remain unchanged for* QRD, DRP *and* RDC.

The data complexity bounds of Theorem 8.3 remain unchanged for F_{MS} *and* F_{MM}. *In contrast, for* F_{mono} *and for CQ, UCQ, $\exists$FO$^+$and FO,*

— QRD *is* NP-*complete;*
— DRP *is* coNP-*complete; and*
— RDC *is* #P-*complete under parsimonious reductions.* □

When k is a predefined constant. In contrast, compatibility constraints do not complicate the analyses of query result diversification when the bound k is a constant.

Table I. Combined complexity and data complexity ($^{(*)}$: known for the lower bound)

Objective functions	Languages	Problems		
		QRD($\mathcal{L}_Q, F$) (Th. 5.1, 5.2, 5.4)	DRP($\mathcal{L}_Q, F$) (Th. 6.1, 6.2, 6.4)	RDC($\mathcal{L}_Q, F$) (Th. 7.1, 7.2, 7.4,7.5)
		Combined complexity		
F_{MS} and F_{MM}	CQ, UCQ, $\exists$FO$^+$	NP-complete$^{(*)}$	coNP-complete	#·NP-complete
	FO	PSPACE-complete	PSPACE-complete	#·PSPACE-complete
F_{mono}	CQ, UCQ, $\exists$FO$^+$, FO	PSPACE-complete	PSPACE-complete	#·PSPACE-complete
		Data complexity		
F_{MS} and F_{MM}	CQ,UCQ,$\exists$FO$^+$,FO	NP-complete$^{(*)}$	coNP-complete	#P-complete (parsimonious)
F_{mono}	CQ,UCQ,$\exists$FO$^+$,FO	PTIME	PTIME	#P-complete (Turing)

COROLLARY 9.7. *For a predefined constant k, in the presence of compatibility constraints of $\mathcal{C}_m$, the combined complexity and data complexity of Corollary 8.4 remain unchanged for* QRD, DRP *and* RDC*, respectively.* □

Summary. From the results of this section, we find the impact of compatibility constraints of $\mathcal{C}_m$ on the complexity of query results diversification as follows.

(1) Although the compatibility constraints of $\mathcal{C}_m$ can be validated in PTIME, their presence complicates the data complexity analyses of QRD($\mathcal{L}_Q, F$), DRP($\mathcal{L}_Q, F$) and RDC($\mathcal{L}_Q, F$), to an extent. More specifically, when these problems are tractable in the absence of the constraints, they become intractable when the constraints are present. The impact is particularly evident when F is F_{mono} (Theorem 9.3), or when $\lambda = 1$ and F is either F_{MS} or F_{MM} (Corollary 9.5).

(2) When it comes to the combined complexity, the presence of compatibility constraints makes QRD($\mathcal{L}_Q, F$), DRP($\mathcal{L}_Q, F$) and RDC($\mathcal{L}_Q, F$) harder when F is F_{mono} and when $\mathcal{L}_Q$ consists of identity queries only (Corollary 9.4). The constraints have no impact on the combined complexity analyses when F is F_{MS} or F_{MM}.

(3) When the bound k on the cardinality $|U|$ of selected sets U is a constant, the complexity results are quite robust: both the combined complexity and data complexity of QRD($\mathcal{L}_Q, F$), DRP($\mathcal{L}_Q, F$) and RDC($\mathcal{L}_Q, F$) remain intact no matter whether compatibility constraints of $\mathcal{C}_m$ are present or absent (Corollary 9.7).

10. CONCLUSIONS

We have extended the result diversification model of [Gollapudi and Sharma 2009] by incorporating queries Q, without assuming the entire set $Q(D)$ of query answers as input. We have identified three decision and counting problems in connection with query result diversification, namely, QRD($\mathcal{L}_Q, F$), DRP($\mathcal{L}_Q, F$) and RDC($\mathcal{L}_Q, F$). We have established the upper and lower bounds of these problems, all matching, for both combined complexity and data complexity, when the query language $\mathcal{L}_Q$ is CQ, UCQ, $\exists$FO$^+$or FO, and when F ranges over all three objective functions F_{MS}, F_{MM} and F_{mono} given in [Gollapudi and Sharma 2009]. We have also studied special cases of these problems, and identified tractable cases. In addition, we have investigated the impact of compatibility constraints on the analyses of query result diversification.

The main complexity results are summarized in Table I, annotated with their corresponding theorems. The complexity bounds of special cases are shown in Table II, followed by Table III for the complexity results in the presence of compatibility constraints that differ from their counterparts in the absence of constraints. The tables

Table II. Special cases

Conditions	Complexity	Problems		
		QRD$(\mathcal{L}_Q, F)$	DRP$(\mathcal{L}_Q, F)$	RDC$(\mathcal{L}_Q, F)$
Identity queries; F is F_{mono}	Combined (Cor. 8.1)	PTIME	PTIME	#P-complete (Turing)
$\lambda = 0$; F is F_{MS}	Data (Th. 8.2)	PTIME	PTIME	#P-complete (Turing)
$\lambda = 0$; F is F_{MM}	Data (Th. 8.2)	PTIME	PTIME	FP
$\lambda = 0$; CQ, UCQ, $\exists$FO$^+$; F is F_{mono};	Combined (Th. 8.2)	NP-complete	coNP-complete	#·NP-complete
k is a constant; F is F_{MS},F_{MM} or F_{mono}	Data (Cor. 8.4)	PTIME	PTIME	FP

Table III. Complexity results in the presence of compatibility constraints

Conditions	Complexity	Problems		
		QRD$(\mathcal{L}_Q, F)$	DRP$(\mathcal{L}_Q, F)$	RDC$(\mathcal{L}_Q, F)$
F is F_{mono}	Data (Th. 9.3)	NP-complete	coNP-complete	#P-complete (parsimonious)
Identity queries; F is F_{mono}	Combined / Data (Cor. 9.4)	NP-complete	coNP-complete	#P-complete (parsimonious)
$\lambda = 0$; F is F_{MS},F_{MM},F_{mono}	Data (Cor. 9.5)	NP-complete	coNP-complete	#P-complete (parsimonious)
$\lambda = 1$; F is F_{mono}	Data (Cor. 9.6)	NP-complete	coNP-complete	#P-complete (parsimonious)

tell us the impact of various factors on the complexity of diversification analyses, such as query languages $\mathcal{L}_Q$, objective functions F, relevance and distance functions δ_{rel} and δ_{dis}, bound k on the number of answers, and compatibility constraints. As annotated in Table I, among all these results, only the NP lower bound of QRD$(\mathcal{L}_Q, F)$ was known prior to this work, when F is F_{MS} or F_{MM}, for CQ, UCQ and $\exists$FO$^+$.

Several extensions are targeted for future work. First, diversification analyses are mostly intractable. We need to identify more special cases that are practical and tractable. Second, we need to develop heuristic algorithms (approximation whenever possible) for those intractable cases. Third, the study should be extended to other objective functions. Fourth, we have only considered a simple class $\mathcal{C}_m$ of compatibility constraints that can be validated in PTIME. More expressive constraint languages should be developed if the need for such languages emerges from practice. Finally, in practice one may want to incorporate user preferences [Chen and Li 2007; Stefanidis et al. 2010] into the diversification model. While we may encode certain preferences in, *e.g.,* the relevance and distance functions, this issue deserves a full treatment.

REFERENCES

ABITEBOUL, S., HULL, R., AND VIANU, V. 1995. *Foundations of Databases*. Addison-Wesley.

ADOMAVICIUS, G. AND TUZHILIN, A. 2005. Towards the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. and Data Eng. 17,* 6, 734–749.

AGRAWAL, R., GOLLAPUDI, S., HALVERSON, A., AND LEONG, S. 2009. Diversifying search results. In *Proc. Int. Conf. Web Search and Web Data Mining*.

AMER-YAHIA, S. 2011. Recommendation projects at Yahoo! *IEEE Data Eng. Bull. 34,* 2, 69–77.

AMER-YAHIA, S., BONCHI, F., CASTILLO, C., FEUERSTEIN, E., MÉNDEZ-DÍAZ, I., AND ZABALA, P. 2013. Complexity and algorithms for composite retrieval. In *Proc. Int. Conf. on World Wide Web*.

BERBEGLIA, G. AND HAHN, G. 2010. Counting feasible solutions of the traveling salesman problem with pickups and deliveries is #P-complete. *Discrete Applied Mathematics 157,* 11, 2541–2547.

BORODIN, A., LEE, H. C., AND YE, Y. 2012. Max-sum diversification, monotone submodular functions and dynamic updates. In *Proc. ACM SIGACT-SIGMOD Symp. on Principles of Database Systems*.

CAPANNINI, G., NARDINI, F. M., PEREGO, R., AND SILVESTRI, F. 2011. Efficient diversification of Web search results. In *Proc. Int. Conf. on Very Large Data Bases*.

CHEN, Z. AND LI, T. 2007. Addressing diverse user preferences in SQL-query-resul navigation. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*.

DEMIDOVA, E., FANKHAUSER, P., ZHOU, X., AND NEJDL, W. 2010. DivQ: Diversification for keyword search over structured databases. In *Proc. Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*.

DENG, T., FAN, W., AND GEERTS, F. 2012. On the complexity of package recommendation problems. In *Proc. ACM SIGACT-SIGMOD Symp. on Principles of Database Systems*.

DROSOU, M. AND PITOURA, E. 2009. Diversity over continuous data. *IEEE Data Eng. Bull. 32,* 4.

DROSOU, M. AND PITOURA, E. 2010. Search result diversification. *SIGMOD Record 39,* 1, 41–47.

DURAND, A., HERMANN, M., AND KOLAITIS, P. G. 2005. Subtractive reductions and complete problems for counting complexity classes. *Theor. Comp. Sci. 340,* 3, 496–513.

FAGIN, R., LOTEM, A., AND NAOR, M. 2003. Optimal aggregation algorithms for middleware. *J. Comp. and System Sci. 66,* 4, 614–656.

FEUERSTEIN, E., HEIBER, P. A., MARTÍNEZ-VIADEMONTE, J., AND BAEZA-YATES, R. A. 2007. New stochastic algorithms for scheduling ads in sponsored search. In *Proc. Latin American Web Congress*.

FRATERNALI, P., MARTINENGHI, D., AND TAGLIASACCHI, M. 2012. Top-k bounded diversification. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*.

GOLLAPUDI, S. AND SHARMA, A. 2009. An axiomatic approach for result diversification. In *Proc. Int. Conf. on World Wide Web*.

HEMASPAANDRA, L. A. AND VOLLMER, H. 1995. The satanic notations: Counting classes beyond #P and other definitional adventures. *SIGACT News 26,* 1, 2–13.

ILYAS, I. F., BESKALES, G., AND SOLIMAN, M. A. 2008. A survey of top-k query processing techniques in relational database systems. *ACM Comput. Surv. 40,* 4, 11:1–11:58.

JIN, W. AND PATEL, J. M. 2011. Efficient and generic evaluation of ranked queries. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*.

KOUTRIKA, G., BERCOVITZ, B., AND GARCIA-MOLINA, H. 2009. FlexRecs: expressing and combining flexible recommendations. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*.

LADNER, R. E. 1989. Polynomial space counting problems. *SIAM J. Comput. 18,* 6, 1087–1097.

LAPPAS, T., LIU, K., AND TERZI, E. 2009. Finding a team of experts in social networks. In *Proc. Int. Conf. on Knowledge Discovery and Data Mining*.

LI, C., SOLIMAN, M. A., CHANG, K. C.-C., AND ILYAS, I. F. 2005. RankSQL: Supporting ranking queries in relational database management systems. In *Proc. Int. Conf. on Very Large Data Bases*.

LIU, Z., SUN, P., AND CHEN, Y. 2009. Structured search result differentiation. In *Proc. Int. Conf. on Very Large Data Bases*.

MINACK, E., DEMARTINI, G., AND NEJDL, W. 2009. Current approaches to search result diversification. In *Proc. Int. Workshop on Living Web*.

PAPADIMITRIOU, C. H. 1994. *Computational Complexity*. Addison-Wesley.

PARAMESWARAN, A. G., GARCIA-MOLINA, H., AND ULLMAN, J. D. 2010. Evaluating, combining and generalizing recommendations with prerequisites. In *Proc. Int. Conf. on Information and Knowledge Management*.

PARAMESWARAN, A. G., VENETIS, P., AND GARCIA-MOLINA, H. 2011. Recommendation systems with complex constraints: A course recommendation perspective. *ACM Trans. Information Syst. 29,* 4.

PROKOPYEV, O. A., KONG, N., AND MARTINEZ-TORRES, D. L. 2009. The equitable dispersion problem. *European Journal of Operational Research 197,* 1, 59–67.

SCHNAITTER, K. AND POLYZOTIS, N. 2008. Evaluating rank joins with optimal cost. In *Proc. ACM SIGACT-SIGMOD Symp. on Principles of Database Systems*.

STEFANIDIS, K., DROSOU, M., AND PITOURA, E. 2010. Perk: Personalized keyword search in relational databases through preferences. In *Proc. Int. Conf. on Extending Database Technology*.

VALIANT, L. 1979. The complexity of computing the permanent. *Theoretical Computer Science 8,* 2, 189–201.

VARDI, M. Y. 1982. The complexity of relational query languages. In *Proc. Annual ACM Symp. on Theory of Computing*.

VEE, E., SRIVASTAVA, U., SHANMUGASUNDARAM, J., BHAT, P., AND YAHIA, S. A. 2008. Efficient computation of diverse query results. In *Proc. Int. Conf. on Data Engineering*.

VIEIRA, M. R., RAZENTE, H. L., BARIONI, M. C. N., HADJIELEFTHERIOU, M., SRIVASTAVA, D., JR., C. T., AND TSOTRAS, V. J. 2011. On query result diversification. In *Proc. Int. Conf. on Data Engineering*.

XIE, M., LAKSHMANAN, L. V. S., AND WOOD, P. T. 2012. Composite recommendations: From items to packages. *Frontiers of Computer Science 6,* 3, 264–277.

YU, C., LAKSHMANAN, L., AND AMER-YAHIA, S. 2009a. It takes variety to make a world: Diversification in recommender systems. In *Proc. Int. Conf. on Extending Database Technology*. 368–378.

YU, C., LAKSHMANAN, L. V., AND AMER-YAHIA, S. 2009b. Recommendation diversification using explanations. In *Proc. Int. Conf. on Data Engineering*.

ZHANG, M. AND HURLEY, N. 2008. Avoiding monotony: Improving the diversity of recommendation lists. In *Proc. ACM Conf. on Recommender Systems*.

ZIEGLER, C.-N., MCNEE, S. M., KONSTAN, J. A., AND LAUSEN, G. 2005. Improving recommendation lists through topic diversification. In *Proc. Int. Conf. on World Wide Web*.

Online Appendix to: On the Complexity of Query Result Diversification

TING DENG, RCBD and SKLSDE, Beihang University
WENFEI FAN, Informatics, University of Edinburgh, and RCBD and SKLSDE, Beihang University

A. PROOFS OF SECTION 8

COROLLARY 8.1. *For identity queries, the combined complexity and data complexity of* QRD, DRP *and* RDC *coincide. More specifically,*

— $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{MS}})$ *and* $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{MM}})$ *are* NP*-complete,*
— $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{MS}})$ *and* $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{MM}})$ *are* coNP*-complete, and*
— $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{MS}})$ *and* $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{MM}})$ *are* #P*-complete under parsimonious reductions,*

for both combined complexity and data complexity, while

— $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{mono}})$ *is in* PTIME*,*
— $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{mono}})$ *is in* PTIME*, and*
— $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$ *is* #P*-complete under polynomial Turing reductions,*

for both combined complexity and data complexity, which are the same as their data complexity given in Theorems 5.4, 6.4 and 7.5, respectively. □

PROOF. For identity queries, we first study the combined and data complexity of QRD, DRP and RDC for F_{MS} or F_{MM}. We then investigate these problems for F_{mono}

(1) When F is F_{MS} or F_{MM}. The data complexity proofs of Theorems 5.4, 6.4 and 7.4 for $\mathsf{QRD}(\mathcal{L}_Q, F)$, $\mathsf{DRP}(\mathcal{L}_Q, F)$ and $\mathsf{RDC}(\mathcal{L}_Q, F)$ when F is F_{MS} or F_{MM} use a fixed identity query as Q. Hence the lower bounds hold here. Moreover, for the upper bound, Theorems 5.1 and 6.1 tell us that $\mathsf{QRD}(\mathcal{L}_Q, F)$ and $\mathsf{DRP}(\mathcal{L}_Q, F)$ are in NP and coNP, respectively, for CQ. Since CQ subsumes identity queries, $\mathsf{QRD}(\mathcal{L}_Q, F)$ and $\mathsf{DRP}(\mathcal{L}_Q, F)$ are in NP and coNP, respectively, for identity queries. Furthermore, the proof of Theorem 7.1 shows that if $Q(D)$ is PTIME computable, such as when Q is an identity query, the problem for verifying whether a given set is valid for (Q, D, k, F, B) is in PTIME. Hence, $\mathsf{RDC}(\mathcal{L}_Q, F)$ is in #·P (*i.e.,* #P) for identity queries.

(2) When F is F_{mono}. Consider the PTIME-algorithms given in the proofs of Theorems 5.4 and 6.4 for the data complexity analyses of $\mathsf{QRD}(\mathsf{FO}, F_{\mathsf{mono}})$ and $\mathsf{DRP}(\mathsf{FO}, F_{\mathsf{mono}})$, respectively. Note that $Q(D)$ and F_{mono} are PTIME computable when Q is an identity query. Thus these PTIME algorithms also work here, and their combined complexity and data complexity coincide to be in PTIME for identity queries.

We next consider $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$. It suffices to show that $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$ is #P-hard for fixed identity queries, and that it is in #P for identity queries that are not necessarily fixed. Observe the following. (a) The lower bound proof of Theorem 7.5 for $\mathsf{RDC}(\mathsf{CQ}, F_{\mathsf{mono}})$ uses a fixed identity query as Q. Thus the lower bound holds here. (b) It is in PTIME to check whether a given set is valid for $(Q, D, k, F_{\mathsf{mono}}, B)$ as $Q(D)$ and F_{mono} are PTIME computable for identity queries. Thus $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$ is in #P.

This completes the proof of Corollary 8.1. □

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

THEOREM 8.2. *When λ = 0, For F_{MS} and F_{MM}, the combined complexity bounds of* QRD, DRP *and* RDC *remain the same as their counterparts given in Theorems 5.1, 6.1 and 7.1, respectively. In contrast, when $\mathcal{L}_Q$ is CQ, UCQ, $\exists$FO$^+$or FO, the data complexity bounds of these problems are*

— *in* PTIME *for* $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{MS}})$ *and* $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{MM}})$*,*
— *in* PTIME *for* $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{MS}})$ *and* $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{MM}})$*, and*
— #P*-complete for* $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{MS}})$ *under polynomial Turing reductions, but in* FP *for* $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{MM}})$.

For F_{mono}, the combined complexity becomes

— NP*-complete for* $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{mono}})$ *when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$, and* PSPACE*-complete when $\mathcal{L}_Q$ is FO;*
— coNP*-complete for* $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{mono}})$ *when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$, and* PSPACE*-complete for FO; and*
— #·NP*-complete for* $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$ *when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$, and* #·PSPACE*-complete for FO.*

The data complexity bounds of these problems remain the same as their counterparts given in Theorems 5.4, 6.4, 7.4 and 7.5, respectively, when $\mathcal{L}_Q$ is CQ, UCQ, $\exists$FO$^+$or FO. □

PROOF. When $\lambda = 0$, we first study $\mathsf{QRD}(\mathcal{L}_Q, F)$, $\mathsf{DRP}(\mathcal{L}_Q, F)$ and $\mathsf{RDC}(\mathcal{L}_Q, F)$ for F_{MS} and F_{MM}. We then investigate them when F is F_{mono}.

<u>(1) When F is F_{MS} or F_{MM}.</u> We start with the combined complexity analyses.

(1.1) Combined complexity. In these settings, we first prove that $\mathsf{QRD}(\mathcal{L}_Q, F)$, $\mathsf{DRP}(\mathcal{L}_Q, F)$ and $\mathsf{RDC}(\mathcal{L}_Q, F)$ are NP-complete, coNP-complete and #·NP-complete for CQ, UCQ and $\exists$FO$^+$, respectively. We then show that they are PSPACE-complete, PSPACE-complete and #·PSPACE-complete for FO, respectively.

(1.1.1) When $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$.

(A) $\mathsf{QRD}(\mathcal{L}_Q, F)$. It suffices to show that $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{MS}})$ and $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{MM}})$ are NP-hard for CQ, and that they are in NP for $\exists$FO$^+$.

Lower bound. We show that $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MS}})$ and $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MM}})$ are NP-hard by reductions from the 3SAT problem, even when $\lambda = 0$ and k is a constant.

We first consider $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MS}})$. Given an instance φ of 3SAT over variables $\{x_1, \ldots, x_m\}$, we define a database D, a CQ query Q, a real number B, a positive integer k and two functions δ_{rel} and δ_{dis} (for F_{MS}), such that φ is satisfiable if and only if there exists a valid set U for (Q, D, k, F_{MS}, B). In particular, we take $k = 2$ and $B = 1$. That is, U consists of two tuples only and $F_{\mathsf{MS}}(U)$ must be no less than 1.

(1) The database D is specified by a single relation schema R_{01}, with its corresponding instance $I_{01} = \{(1), (0)\}$, encoding the Boolean domain.

(2) We define the query Q in CQ as follows:

$$Q(\vec{x}) = R_{01}(x_1) \wedge \ldots \wedge R_{01}(x_m).$$

Here $\vec{x} = (x_1, \ldots, x_m)$ and Q generates all truth assignments of X variables. Let R_Q denote the schema of query result $Q(D)$.

(3) For each tuple t of R_Q, we define $\delta_{\mathsf{rel}}(t, Q) = 1$ if the truth assignment μ_X encoded by tuple t makes φ true, and let $\delta_{\mathsf{rel}}(t, Q) = 0$ otherwise. Furthermore, we take δ_{dis} as a constant function that returns 0 for each pair of tuples t and t' of R_Q. We set $\lambda = 0$. Then for each set U of k tuples of R_Q, $F_{\mathsf{MS}}(U) = (k-1) \cdot \sum_{t \in U} \delta_{\mathsf{rel}}(t, Q)$ (see Section 3). That is, F_{MS} is defined in terms of δ_{rel} alone (and hence we use $k = 2$).

We show that φ is satisfiable if and only if there is a valid set U for $(Q, D, k, F_{\mathsf{MS}}, B)$.

$\boxed{\Rightarrow}$ First assume that φ is satisfiable. Then there exists a truth assignment μ_X^0 of X variables satisfying φ. Let $U = \{t_0, t\}$, where tuple t_0 encodes μ_X^0 and t is an arbitrary tuple in $Q(D)$ that is distinct from t_0. Obviously, $U \subseteq Q(D)$, $|U| = 2$, and moreover, $F(U) \geq 1 = B$ since $\delta_{\mathsf{rel}}(t_0, Q) = 1$. That is, U is a valid set for $(Q, D, k, F_{\mathsf{MS}}, B)$.

$\boxed{\Leftarrow}$ Conversely, assume that φ is not satisfiable. Then for any tuple t of R_Q, $\delta_{\mathsf{rel}}(t, Q) = 0$ by the definition of δ_{rel}. Thus for each set U of two tuples t and t' of R_Q, $F_{\mathsf{MS}}(U) = 0 < B$ by the definition of F_{MS}. Hence there exists no valid set for $(Q, D, k, F_{\mathsf{MS}}, B)$.

For $\mathsf{QRD}(\mathsf{CQ}, F_{\mathsf{MM}})$, given an instance φ of 3SAT, we construct the same D, Q, δ_{rel}, δ_{dis} as their counterparts given above, and let $k = 1$ and $B = 1$. Furthermore, for each set U of tuples of R_Q, we set $\lambda = 0$ and hence have that $F_{\mathsf{MM}}(U) = \min_{t \in U} \delta_{\mathsf{rel}}(t, Q)$. Then along the same lines as above, one can readily verify that φ is satisfiable if and only if there exists a valid set U for $(Q, D, k, F_{\mathsf{MM}}, B)$.

Upper bound. The algorithms given in the proof of Theorem 5.1 remain intact in the special case when $\lambda = 0$. Thus $\mathsf{QRD}(\exists\mathrm{FO}^+, F_{\mathsf{MS}})$ and $\mathsf{QRD}(\exists\mathrm{FO}^+, F_{\mathsf{MM}})$ are in NP.

(B) $\mathsf{DRP}(\mathcal{L}_Q, F)$. It suffices to show that $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{MS}})$ and $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{MM}})$ are coNP-hard for CQ and that they are in coNP for $\exists\mathrm{FO}^+$.

Lower bound. We verify that $\mathsf{DRP}(\mathsf{CQ}, F_{\mathsf{MS}})$ and $\mathsf{DRP}(\mathsf{CQ}, F_{\mathsf{MM}})$ are coNP-hard, when $\lambda = 0$ and k is a constant, by reductions from the complement of 3SAT, which is known to be coNP-complete (cf. [Papadimitriou 1994]).

We first study $\mathsf{DRP}(\mathsf{CQ}, F_{\mathsf{MS}})$. Given an instance $\varphi = C_1 \wedge \ldots \wedge C_l$ of 3SAT over variables X, we define a database D, a CQ query Q, functions δ_{rel}, δ_{dis} and F_{MS}, a set $U \subseteq Q(D)$, and a positive integer k. We prove that $\mathsf{rank}(U) \leq r$ if and only if φ is not satisfiable. In particular, we set $k = 2$ and $r = 1$. That is, the set U consists of two tuples only and has the highest rank.

Before giving the reduction, we first define $\varphi' = (\varphi \vee z) \wedge \bar{z} = \bigwedge_{i=1}^{l} (C_i \vee z) \wedge \bar{z}$, where z is a fresh variable that is not in the set X of variables in φ. As discussed in the proof of Theorem 6.1 for $\mathsf{DRP}(\mathsf{CQ}, F_{\mathsf{MS}})$, for a truth assignment μ_X of X variables, μ_X satisfies φ if and only if μ_X makes φ' true with $z = 0$, and moreover, when setting z to be 1, φ' is false under any truth assignments in X.

We next give the reduction as follows.

(1) The database consists of four relations I_{01}, $I_\vee$, $I_\wedge$ and $I_\neg$ as shown in Fig. 5, specified by schemas $R_{01}(X)$, $R_\vee(B, A_1, A_2)$, $R_\wedge(B, A_1, A_2)$ and $R_\neg(A, \bar{A})$, respectively. Here I_{01} encodes the Boolean domain, and $I_\vee$, $I_\wedge$ and $I_\neg$ encode disjunction, conjunction and negation, respectively, such that φ and φ' can be expressed in CQ with these relations.

(2) We define the CQ query Q as follows:

$$Q(b, c) \ = \ \exists \vec{x}\, \exists z \big((Q_X(\vec{x}\,) \ \wedge Q_{\varphi'}(\vec{x}, z, b)) \ \wedge R_{01}(c) \big).$$

Here $\vec{x} = (x_1, \ldots, x_m)$. Query $Q_X(\vec{x}\,)$ generate all truth assignments of X variables, by means of Cartesian products of R_{01}. The sub-query $Q_{\varphi'}(\vec{x}, z, b)$ encodes the truth value of φ' (*i.e.,* b), for a given truth assignment μ_X represented by $\vec{x}$ and truth assignment μ_z for z, such that $b = 1$ if (μ_X, μ_z) satisfies φ', and $b = 0$ otherwise. Obviously, $Q_{\varphi'}(\vec{x}, z, b)$ can be expressed in CQ in terms of $R_\vee$, $R_\wedge$, and $R_\neg$. Observe that given D, $Q(D)$ is a subset of $\{(1,1),\ (1,0),\ (0,1),\ (0,0)\}$. Let $U = \{(0,1),\ (0,0)\}$. As remarked above, φ' is false under the truth assignments when $z = 1$; hence we have that $U \subseteq Q(D)$.

(3) We define $\delta_{\mathsf{rel}}((1,0), Q) = \delta_{\mathsf{rel}}((1,1), Q) = 2$ and $\delta_{\mathsf{rel}}((0,1), Q) = \delta_{\mathsf{rel}}((0,0), Q) = 1$. Furthermore, we use a constant function δ_{dis} that returns 0 for each pair of tuples t and s of

R_Q. We set $\lambda = 0$. Then for each set S of k tuples of R_Q, $F_{\mathsf{MS}}(S) = (k-1) \cdot \sum_{t \in S} \delta_{\mathsf{rel}}(t, Q)$ (see Section 3; recall $k = 2$). Thus $F_{\mathsf{MS}}(\{(1,1),(1,0)\}) = 4$, $F_{\mathsf{MS}}(\{(0,1),(0,0)\}) = 2$, and $F_{\mathsf{MS}}(\{t,s\}) = 3$ if $t \in \{(1,1),(1,0)\}$ and $s \in \{(0,0),(0,1)\}$.

We next show that φ is not satisfiable if and only if $\mathsf{rank}(U) \leq r$.

$\boxed{\Rightarrow}$ Assume that φ is not satisfiable. Then there exists no truth assignment μ_X of X variables that satisfies φ. Thus, tuples $(1,1)$ and $(1,0)$ cannot be in the answer $Q(D)$ to query Q in D. Therefore, $\mathsf{rank}(U) = 1 \leq r$, by the definition of F_{MS}.

$\boxed{\Leftarrow}$ Conversely, assume that φ is satisfiable. Then there exists a truth assignment μ_X^0 of X variables that satisfies φ. Thus $(1,1)$ and $(1,0)$ must be in $Q(D)$, by the definition of query Q. Hence $\mathsf{rank}(U) > 1 = r$, by the definition of F_{MS}.

We next show that $\mathsf{DRP}(\mathsf{CQ}, F_{\mathsf{MM}})$ is coNP-hard, also by reduction from the complement of 3SAT. Given an instance φ of 3SAT, we construct the same φ', D, Q, δ_{rel}, and δ_{dis} as their counterparts for F_{MS} given above, and let $U = \{(0,1)\}$ and $k = r = 1$. Furthermore, we set $\lambda = 0$. Then for each set S of tuples of R_Q, $F_{\mathsf{MM}}(S) = \min_{t \in S} \delta_{\mathsf{rel}}(t, Q)$. Then along the same lines as the argument for F_{MS} given above, one can easily verify that $\mathsf{rank}(U) \leq r$ if and only if φ is not satisfiable.

Upper bound. The algorithms given in the proof of Theorem 6.1 obviously work in the special case when $\lambda = 0$. Thus $\mathsf{QRD}(\exists\mathrm{FO}^+, F_{\mathsf{MS}})$ and $\mathsf{QRD}(\exists\mathrm{FO}^+, F_{\mathsf{MM}})$ are in coNP.

(C) $\mathsf{RDC}(\mathcal{L}_Q, F)$. Recall the lower bounds of $\mathsf{RDC}(\mathcal{L}_Q, F)$ given in Theorems 7.1 for F_{MS} and F_{MM} when $\mathcal{L}_Q$ ranges over CQ, UCQ or $\exists\mathrm{FO}^+$. Those bounds are established by taking $\lambda = 0$ and hence, hold here. For the upper bounds, the algorithms given there obviously remain intact in the special case when $\lambda = 0$.

(1.1.2) When $\mathcal{L}_Q$ is FO. The lower bounds of $\mathsf{QRD}(\mathrm{FO}, F)$, $\mathsf{DRP}(\mathrm{FO}, F)$ and $\mathsf{RDC}(\mathrm{FO}, F)$ given in Theorems 5.1, 6.1 and 7.1 for F_{MS} and F_{MM} are established by taking $\lambda = 0$. As a result, those lower bounds hold here. For the upper bounds, the algorithms given there obviously remain intact in the special case when $\lambda = 0$.

(1.2) Data complexity. It suffices to show that $\mathsf{QRD}(\mathrm{FO}, F)$ and $\mathsf{DRP}(\mathrm{FO}, F)$ are in PTIME when F is F_{MS} or F_{MM}, $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{MS}})$ is #P-complete under polynomial Turing reductions for CQ, UCQ, $\exists\mathrm{FO}^+$and FO, and $\mathsf{RDC}(\mathrm{FO}, F_{\mathsf{MM}})$ is in FP.

(1.2.1) QRD*(FO,* F_{MS}). We develop a PTIME algorithm for $\mathsf{QRD}(\mathrm{FO}, F_{\mathsf{MS}})$ when $\lambda = 0$. Recall that $F_{\mathsf{MS}}(U) = (k-1) \cdot \sum_{t \in U} \delta_{\mathsf{rel}}(t, Q)$ for each set U of k tuples of schema R_Q in this setting. Hence we develop an algorithm that works as follows:

1. compute $Q(D)$ and sort the tuples in $Q(D)$ in descending order based on δ_{rel};
2. check whether $|Q(D)| \geq k$; if so, continue; otherwise, return "no";
3. let U be the set consisting of the first k tuples in the sorted $Q(D)$; check whether $F_{\mathsf{MS}}(U) \geq B$; if so, return "yes"; otherwise, return "no".

It is easy to verify that the algorithm is correct. Moreover, the algorithm is in PTIME. Indeed, steps 1 and 2 are in PTIME since it is in PTIME to compute $Q(D)$ for a fixed FO query Q, and step 3 is in PTIME because F_{MS} is PTIME computable in this setting.

(1.2.2) $\mathsf{QRD}(\mathit{FO}, F_{\mathsf{MM}})$. When $\lambda = 0$, $F_{\mathsf{MM}}(U) = \min_{t \in U} \delta_{\mathsf{rel}}(t, Q)$, and it is PTIME computable. It is easy to see that the PTIME algorithm for $\mathsf{QRD}(\mathrm{FO}, F_{\mathsf{MS}})$ given above also works here. Hence $\mathsf{QRD}(\mathrm{FO}, F_{\mathsf{MM}})$ is also in PTIME.

(1.2.3) $\mathsf{DRP}(\mathit{FO}, F_{\mathsf{MS}})$ *and* $\mathsf{DRP}(\mathit{FO}, F_{\mathsf{MM}})$. When $\lambda = 0$, for each k-tuple set U of schema R_Q of $Q(D)$, $F_{\mathsf{MS}}(U) = (k-1) \cdot \sum_{t \in U} \delta_{\mathsf{rel}}(t, Q)$, and $F_{\mathsf{MM}}(U) = \min_{t \in U} \delta_{\mathsf{rel}}(t, Q)$. Recall the PTIME-algorithm given in the proof of Theorem 6.4 for $\mathsf{DRP}(\mathrm{FO}, F_{\mathsf{mono}})$. Obviously,

if we define $v(t) = \delta_{\mathsf{rel}}(t, Q)$ for each tuple $t \in Q(D)$ and use F_{MS} instead of F_{mono} in the algorithm, the algorithm can be used for $\mathsf{DRP}(\mathrm{FO}, F_{\mathsf{MS}})$, and is still in PTIME. Similarly, when using function v given above and F_{MM} instead of F_{mono}, the algorithm also works for $\mathsf{DRP}(\mathrm{FO}, F_{\mathsf{MM}})$, and is also in PTIME.

(1.2.4) $\mathsf{RDC}(\mathit{FO}, F_{\mathsf{MS}})$. It suffices to show that $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{MS}})$ is #P-hard under polynomial Turing reductions and that $\mathsf{RDC}(\mathrm{FO}, F_{\mathsf{MS}})$ is in #P.

lower bound. We show that $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{MS}})$ is #P-hard by polynomial Turing reduction from #SSPk, which is #P-complete by Lemma 7.6. Along the same line as the proof of Theorem 7.5 for $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{mono}})$, we construct a polynomial Turing reduction as follows. Given an instance W, π, l and d of #SSPk, we define the same transformation from #SSPk to $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{MS}})$ as given in the proof of Theorem 7.5 for $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{mono}})$, except the following: (a) when $\lambda = 0$, for each set U consisting of k tuples of R_Q, $F_{\mathsf{MS}}(U) = (k-1) \cdot \sum_{t \in U} \delta_{\mathsf{rel}}(t, Q)$; and (b) we let $k = l$ and $B = (l-1) \cdot d$. It is easy to see that the number of subsets T of W with $|T| = l$ and $\sum_{w \in T} \pi(w) \geq d$ equals the number of valid sets U for $(Q, D, k, F_{\mathsf{MS}}, B)$. As discussed there, we can find the solution to #SSPk, *i.e.,* the number of subsets T of W with $|T| = l$ and $\sum_{w \in T} \pi(w) = d$, by calling the oracle $\mathsf{COUNT}_{\mathsf{RDC}}$ *twice*, to compute the numbers X and Y of valid sets for $(Q, D, k, F_{\mathsf{MS}}, B)$ and $(Q, D, k, F_{\mathsf{MS}}, B+1)$, respectively, and the solution to #SSPk is simply $X - Y$. Here $\mathsf{COUNT}_{\mathsf{RDC}}(Q, D, k, F_{\mathsf{MS}}, B)$ is the oracle that given Q, D, k, F_{MS} and B, returns the number of valid sets U for $(Q, D, k, F_{\mathsf{MS}}, B)$.

Upper bound. To see that $\mathsf{RDC}(\mathrm{FO}, F_{\mathsf{MS}})$ is in #P, we only need to show that it is in PTIME to verify whether a given set U is valid for $(Q, D, k, F_{\mathsf{MS}}, B)$. Indeed, $Q(D)$ is PTIME computable since Q is fixed, and moreover, F_{MS} is also PTIME computable.

(1.2.5) $\mathsf{RDC}(\mathit{FO}, F_{\mathsf{MM}})$. When $\lambda = 0$, we show that $\mathsf{RDC}(\mathrm{FO}, F_{\mathsf{MM}})$ is in FP for fixed queries Q, by giving an FP algorithm. Given Q, D, k, F_{MM}, and B, the algorithm returns the number of valid sets for $(Q, D, k, F_{\mathsf{MM}}, B)$. Recall that $F_{\mathsf{MM}}(U) = \min_{t \in U} \delta_{\mathsf{rel}}(t, Q)$ when $\lambda = 0$ (see Section 3). The algorithm works as follows:

1. compute $Q(D)$ and sort tuples in $Q(D)$ in descending order based on their δ_{rel} values; let $Q(D) = \{t_1, \ldots, t_{|Q(D)|}\}$, where $\delta_{\mathsf{rel}}(t_i, Q) \geq \delta_{\mathsf{rel}}(t_j, Q)$ when $i \leq j$;
2. check whether $\delta_{\mathsf{rel}}(t_i, Q) < B$ for all $i \in [1, |Q(D)|]$; if so, return 0; otherwise continue;
3. let t_i be the tuple such that $\delta_{\mathsf{rel}}(t_i, Q) \geq B$ and $\delta_{\mathsf{rel}}(t_{i+1}, Q) < B$; check whether $i \geq k$; if so, return $= \mathrm{C}_i^k$, represented in binary; otherwise return 0.

Here step 1 is in PTIME since $Q(D)$ is PTIME computable when Q is fixed. Step 2 is obviously in PTIME when $\lambda = 0$. Moreover, step 3 is in PTIME since the output is represented in binary. Thus the algorithm is in PTIME.

(2) When F is F_{mono}. We first study the combined complexity of $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{mono}})$, $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{mono}})$ and $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$. We then consider their data complexity.

(2.1) Combined complexity. Note that when $\lambda = 0$, $F_{\mathsf{mono}}(U) = \sum_{t \in U} \delta_{\mathsf{rel}}(t, Q)$, and $F_{\mathsf{MS}}(U) = (k-1) \cdot \sum_{t \in U} \delta_{\mathsf{rel}}(t, Q)$ for each set U of k tuples of R_Q. As a result, when $\lambda = 0$ and $k = 2$, F_{mono} and F_{MS} are the same function. Recall that in the lower bounds proofs of Theorem 8.2 (for $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{MS}})$ and $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{MS}})$) and Theorem 7.1 (for $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{MS}})$), we set $\lambda = 0$ and $k = 2$. Thus the lower bounds given there hold here for F_{mono}. Moreover, the algorithms given in those upper bound proofs carry over to the special case for $\lambda = 0$. Hence the upper bounds hold here.

(2.2) Data complexity. We show that when $\lambda = 0$, the data complexity bounds of $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{mono}})$, $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{mono}})$ and $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$ remain the same as their counterparts given in Theorems 5.4, 6.4 and 7.5, respectively. Obviously, the PTIME

algorithms given for Theorems 5.4 and 6.4 carry over to the special case when $\lambda = 0$. For $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$, recall that its lower bound proof for Theorem 7.5 uses $\lambda = 0$. Hence the lower bound remains intact here. Moreover, the algorithm given there obviously also works in the special case when $\lambda = 0$.

This completes the proof of Theorem 8.2. □

THEOREM 8.3. *When λ = 1, the combined complexity of Theorems 5.1, 5.2, 6.1, 6.2, 7.1 and 7.2 and the data complexity of Theorems 5.4, 6.4, 7.4 and 7.5 remain unchanged for* QRD, DRP *and* RDC*, respectively.* □

PROOF. When $\lambda = 1$, we first study QRD, DRP and RDC for F_{MS} and F_{MM}. We then investigate these problems when F is F_{mono}.

(1) When F is F_{MS} or F_{MM}. We first study the combined complexity, and then investigate the data complexity of these problems in these settings.

(1.1) Combined complexity. We first consider CQ, UCQ and $\exists$FO$^+$, and then FO.

(1.1.1) When $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$. The lower bounds of $\mathsf{QRD}(\mathsf{CQ}, F)$ and $\mathsf{DRP}(\mathsf{CQ}, F)$ given in Theorems 5.1 and 6.1 for F_{MS} and F_{MM} are established by taking $\lambda = 1$. As a result, these lower bounds hold here. For the upper bounds, the algorithms given there obviously remain intact in the special case when $\lambda = 0$.

We next only need to show that $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{MS}})$ and $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{MM}})$ are #P-complete for CQ, UCQ and $\exists$FO$^+$, when $\lambda = 1$.

Lower bound. We first show that $\mathsf{RDC}(\mathsf{CQ}, F_{\mathsf{MS}})$ is #·NP-hard by parsimonious reduction from $\#\Sigma_1$SAT (see Section 7.1). Given an instance $\varphi(X, Y) = \exists X(C_1 \wedge \ldots \wedge C_l)$ of $\#\Sigma_1$SAT, we use the same reduction given in the proof of Theorem 7.1 for $\mathsf{RDC}(\mathsf{CQ}, F_{\mathsf{MS}})$, except the following: (i) we define $\delta_{\mathsf{dis}}((t_Y, 0, 1), (1, \ldots, 1, 0)) = 1$, and for any other pair of tuples t and s, we define $\delta_{\mathsf{dis}}(t, s) = 0$; (ii) we set $\lambda = 1$ and $k = 2$; hence for each set U of tuples of R_Q, $F_{\mathsf{MS}}(U) = \sum_{t,s \in U} \delta_{\mathsf{dis}}(t, s)$; and (iii) we set $B = 1$. Then one can verify that the number of valid sets for $(Q, D, k, F_{\mathsf{MS}}, B)$ is equal to the number of truth assignments of Y that satisfy φ. This follows from the fact that for each set $U \subseteq Q(D)$ such that $|U| = k = 2$, $F_{\mathsf{MS}}(U) \geq B = 1$ if and only if $U = \{(t_Y, 0, 1), (1, \ldots, 1, 0)\}$, where the truth assignment encoded by t_Y satisfies φ.

We next show that $\mathsf{RDC}(\mathsf{CQ}, F_{\mathsf{MM}})$ is #·NP-hard also by parsimonious reduction from $\#\Sigma_1$SAT. Given an instance $\varphi(X, Y)$ of $\#\Sigma_1$SAT, we use the same reduction as given above for $\mathsf{RDC}(\mathsf{CQ}, F_{\mathsf{MS}})$ except that when $\lambda = 1$, $F_{\mathsf{MM}}(U) = \min_{t,s \in U, t \neq s} \delta_{\mathsf{dis}}(t, s)$ for each set U consisting of k tuples of R_Q. Then along the same line as the proof given above, one can show that the number of valid sets for $(Q, D, k, F_{\mathsf{MM}}, B)$ equals the number of truth assignments of Y that satisfy φ.

Upper bound. It is easy to see that when $\lambda = 1$, it is still in NP to verify whether a given set U is valid for (Q, D, k, F, B) for $\exists$FO$^+$, when F is F_{MS} or F_{MM} following the proof of Theorem 7.1. Hence $\mathsf{RDC}(\exists\mathsf{FO}^+, F)$ is in #·NP in this case.

(1.1.2) When $\mathcal{L}_Q$ is FO. We show that when $\lambda = 1$, $\mathsf{QRD}(\mathsf{FO}, F)$ and $\mathsf{DRP}(\mathsf{FO}, F)$ are PSPACE-complete, and $\mathsf{RDC}(\mathsf{FO}, F)$ is #·PSPACE-complete, for F_{MS} and F_{MM}.

(A) Lower bound. We first prove the lower bounds.

(a) $\mathsf{QRD}(\mathit{FO}, F_{\mathsf{MS}})$. We show that when $\lambda = 1$, $\mathsf{QRD}(\mathsf{FO}, F_{\mathsf{MS}})$ is PSPACE-hard by reduction from the membership problem for FO. Given an instance (Q, D, s) of the membership problem, we use the same reduction as the one given in Theorem 5.1 for $\mathsf{QRD}(\mathsf{FO}, F_{\mathsf{MS}})$, except the following: (i) we define $\delta_{\mathsf{dis}}((s, 0), (s, 1)) = 1$, and for any other

two tuples t and t' of R_Q, $\delta_{\mathsf{dis}}(t,t') = 0$, and (ii) we set $\lambda = 1$, and hence, for each set U of tuples of R_Q, $F_{\mathsf{MS}}(U) = \sum_{t,t' \in U} \delta_{\mathsf{dis}}(t,t')$; and (iii) we set $B = 1$. Recall that $k = 2$.

We next show that $s \in Q(D)$ if and only if there exists a valid set for (Q', D', k, F_{MS}, B). First assume that $s \in Q(D)$. Then there exists a set $U = \{(s,1),(s,0)\}$ valid for (Q', D', k, F_{MS}, B) by the definition of δ_{dis}. Conversely, if $s \notin Q(D)$, then tuples $(s,1)$ and $(s,0)$ are not in $Q(D)$. Thus for each pair of tuples t and t' in $Q(D)$, $F_{\mathsf{MS}}(\{t,t'\}) = 0 < B$.

(b) $\mathsf{QRD}(\textbf{\textit{FO}}, F_{\mathsf{MM}})$. We show that $\mathsf{QRD}(\mathrm{FO}, F_{\mathsf{MM}})$ is PSPACE-hard also by reduction from the membership problem for FO queries. Given an instance (Q,D,s) of the membership problem, we use the same reduction given above for $\mathsf{QRD}(\mathrm{FO}, F_{\mathsf{MS}})$ except that when setting $\lambda = 1$, for each set U of tuples of R_Q, $F_{\mathsf{MM}}(U) = \min_{t,s \in U, t \neq s} \delta_{\mathsf{dis}}(t,s)$. Then along the same line as above, one can readily verify that $s \in Q(D)$ if and only if there exists a valid set U for $(Q', D', k, F_{\mathsf{MM}}, B)$.

(c) $\mathsf{DRP}(\textbf{\textit{FO}}, F_{\mathsf{MS}})$. We show that $\mathsf{DRP}(\mathrm{FO}, F_{\mathsf{MS}})$ is PSPACE-hard by reduction from the complement of the membership problem for FO. Given an instance (Q,D,s) of the membership problem, we use the same reduction given in the proof of Theorem 6.1 for $\mathsf{DRP}(\mathrm{FO}, F_{\mathsf{MS}})$, except the following: (i) we define $\delta_{\mathsf{dis}}((s,1,1),(s,1,0)) = 1$, $\delta_{\mathsf{dis}}((s,0,1),(s,0,0)) = 2$ and for any other two tuples t and t' of R_Q, we let $\delta_{\mathsf{dis}}(t,t') = 0$, and (ii) we set $\lambda = 1$. Recall that $k = 2$, $r = 1$ and $U = \{(s,1,1),(s,1,0)\}$. Hence for each set S of tuples of R_Q, we have that $F_{\mathsf{MS}}(S) = \sum_{t,t' \in S} \delta_{\mathsf{dis}}(t,t')$. It is easy to see that $s \notin Q(D)$ if and only if $\mathsf{rank}(U) = 1 \leq r$.

(d) $\mathsf{DRP}(\textbf{\textit{FO}}, F_{\mathsf{MM}})$. We show that $\mathsf{DRP}(\mathrm{FO}, F_{\mathsf{MM}})$ is PSPACE-hard also by reduction from the complement of the membership problem for FO. Given an instance (Q,D,s) of the membership problem for FO, we use the same reduction given above for $\mathsf{DRP}(\mathrm{FO}, F_{\mathsf{MS}})$, except that when $\lambda = 1$, $F_{\mathsf{MM}}(S) = \min_{t,t' \in S, t \neq t'} \delta_{\mathsf{dis}}(t,t')$ for each set S of tuples of $R_{Q'}$. Then one can verify that $s \notin Q(D)$ if and only if $\mathsf{rank}(U) = 1 \leq r$.

(e) $\mathsf{RDC}(\textbf{\textit{FO}}, F_{\mathsf{MS}})$. We show that when $\lambda = 1$, $\mathsf{RDC}(\mathrm{FO}, F_{\mathsf{MS}})$ is $\#\cdot$PSPACE-hard by parsimonious reduction from #QBF (see Section 7.1). Given an instance $\varphi = \exists X \, \forall y_1 P_2 y_2 \cdots P_n y_n \, \psi$ of #QBF, we use the same reduction given in the proof of Theorem 7.1 for $\mathsf{RDC}(\mathrm{FO}, F_{\mathsf{MS}})$, except the following: (i) we define $\delta_{\mathsf{dis}}((t_X,0,1),(1,\ldots,1,0)) = 1$, where t_X is a truth assignment of the X variables that satisfies φ, and for any other pair of tuples t and t', we define $\delta_{\mathsf{dis}}(t,t') = 0$; (ii) we set $\lambda = 1$ and $k = 2$, and thus for each set U of tuples of R_Q, $F_{\mathsf{MS}}(U) = \sum_{t,t' \in U} \delta_{\mathsf{dis}}(t,t')$; and moreover, (iii) we set $B = 1$. Then along the same line as the proof of Theorem 7.1 for $\mathsf{RDC}(\mathrm{FO}, F_{\mathsf{MS}})$, one can verify that the number of valid sets for $(D,Q,k,F_{\mathsf{MS}},B)$ equals the number of truth assignments of X that satisfy φ.

(f) $\mathsf{RDC}(\textbf{\textit{FO}}, F_{\mathsf{MM}})$. We show that $\mathsf{RDC}(\mathrm{FO}, F_{\mathsf{MM}})$ is $\#\cdot$PSPACE-hard, also by parsimonious reduction from #QBF. Given an instance $\varphi = \exists X \, \forall y_1 P_2 y_2 \cdots P_n y_n \, \psi$ of #QBF, we use the same reduction given above for $\mathsf{RDC}(\mathrm{FO}, F_{\mathsf{MS}})$, except the following: (i) when setting $\lambda = 1$, $F_{\mathsf{MM}}(U) = \min_{t,s \in U, t \neq s} \delta_{\mathsf{dis}}(t,s)$ for each set U of tuples of R_Q; and (ii) we set $B = 1$. Then one can readily verify that the number of valid sets for (D, Q, k, F_{MM}, B) equals the number of truth assignments of X that satisfy φ.

(B) Upper bound. We show that when $\lambda = 1$, $\mathsf{QRD}(\mathrm{FO}, F)$, $\mathsf{DRP}(\mathrm{FO}, F)$ and $\mathsf{RDC}(\mathrm{FO}, F)$ are in PSPACE, PSPACE and $\#\cdot$PSPACE, respectively, when F is F_{MS} or F_{MM}. Obviously, the algorithms given in the proofs of Theorem 5.1 and 6.1 for $\mathsf{QRD}(\mathrm{FO}, F)$ and $\mathsf{DRP}(\mathrm{FO}, F)$, respectively, work here, where F is F_{MS} or F_{MM}. Thus $\mathsf{QRD}(\mathrm{FO}, F)$ and $\mathsf{DRP}(\mathrm{FO}, F)$ are both in PSPACE. Moreover, it is easy to see that when $\lambda = 1$, it is still in PSPACE to verify that whether a set U is a valid set for (Q,D,k,F,B) for FO,

when F is F_{MS} or F_{MM}. Hence $\mathsf{RDC}(\mathrm{FO}, F)$ is in $\#\cdot$PSPACE for max-sum or max-min diversification, when $\lambda = 1$.

(1.2) Data complexity. The lower bounds of $\mathsf{QRD}(\mathcal{L}_Q, F)$, $\mathsf{DRP}(\mathcal{L}_Q, F)$ and $\mathsf{RDC}(\mathcal{L}_Q, F)$ for fixed queries hold here, as their proofs for Theorems 5.4, 6.4, 7.4 and 7.5, respectively, use $\lambda = 1$. The algorithms given there for the upper bounds also work here.

(2) When F is F_{mono}. For the combined complexity, the lower bounds proofs of Theorems 5.2, 6.2 and 7.2 for $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{mono}})$, $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{mono}})$ and $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$, respectively, are verified by using $\lambda = 1$. Hence, those lower bounds hold here. Moreover, all the upper bounds given there carry over to the special case when $\lambda = 1$.

For the data complexity, the PTIME upper bounds of Theorems 5.4 and 6.4 for $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{mono}})$ and $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{mono}})$, respectively, obviously carry over to their special case when $\lambda = 1$. We next show that when $\lambda = 1$, the data complexity of $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$ is #P-complete for CQ, UCQ, $\exists$FO$^+$and FO, under polynomial Turing reductions. It suffices to show that $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{mono}})$ is #P-hard under polynomial Turing reductions, and that $\mathsf{RDC}(\mathrm{FO}, F_{\mathsf{mono}})$ is in #P, for fixed queries.

(2.1) Lower bound. We show that when $\lambda = 1$, $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{mono}})$ is #P-hard by polynomial Turing reduction from #SSPk, which is shown #P-complete by Lemma 7.6. Along the same line as the proof of Theorem 7.5, we construct a polynomial Turing reduction as follows. Given an instance W, π, l and d of #SSPk, we define Q, D, δ_{rel}, δ_{dis}, F_{mono}, k and B, such that the number of subsets T of W with $|T| = l$ and $\sum_{w\in T} \pi(w) \geq d$ equals the number of valid sets U for $(Q, D, k, F_{\mathsf{mono}}, B)$. As discussed there, we can find the solution to #SSPk, *i.e.,* the number of subsets T of W with $|T| = l$ and $\sum_{w\in T} \pi(w) = d$, by calling the oracle $\mathsf{COUNT}_{\mathsf{RDC}}$ twice, to compute the numbers X and Y of valid sets for $(Q, D, k, F_{\mathsf{mono}}, B)$ and $(Q, D, k, F_{\mathsf{mono}}, B + 1)$, respectively. Here $\mathsf{COUNT}_{\mathsf{RDC}}(Q, D, k, F_{\mathsf{mono}}, B)$ is the oracle that given $Q, D, k, F_{\mathsf{mono}}$ and B, returns the number of valid sets U for $(Q, D, k, F_{\mathsf{mono}}, B)$.

We next give the transformation from #SSPk to $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{mono}})$, for $\lambda = 1$.

(1) For each $w \in W$, let w' be a distinct element not in W. The database D consists of a single relation $I_W = \{(w), (w') \mid w \in W\}$, specified by schema $R_W(W)$.

(2) We define query Q as the identity query on R_W instances. Then $|Q(D)| = 2|W|$.

(3) We define δ_{rel} as a constant function that returns 1 for each tuple of R_Q. Moreover, we define $\delta_{\mathsf{dis}}((w), (w')) = \pi(w)$, and for any other pair of tuples t and t' of R_Q, we define $\delta_{\mathsf{dis}}(t, t') = 0$. We set $\lambda = 1$, and thus for each set U of tuples of R_Q, $F_{\mathsf{mono}}(U) = (1/(2|W| - 1)) \cdot \sum_{t\in U, s\in Q(D)} \delta_{\mathsf{dis}}(t, s)$.

(4) Finally, we set $k = 2l$ and $B = d/(2|W| - 1)$.

We only need to show that the number of subsets T of W with $|T| = l$ and $\sum_{w\in T} \pi(w) \geq d$ is equal to the number of valid sets U for $(Q, D, k, F_{\mathsf{mono}}, B)$. Recall that $k = 2l$ and $B = d/(2|W| - 1)$. Assume that there exists a set $U \subseteq Q(D)$ with $|U| = k$ and $F_{\mathsf{mono}}(U) \geq B$. Then by the definition of δ_{dis}, $F_{\mathsf{mono}}(U) = (1/(2|W| - 1)) \cdot \sum_{(w)\in U, (w')\in U} \delta_{\mathsf{dis}}((w), (w')) = (1/(2|W| - 1)) \cdot \sum_{(w)\in U} \pi(w) \geq B$. Then for the set $T = \{w \mid (w) \in U\}$, we have that $\sum_{w\in T} \pi(w) \geq d$. Conversely, for a subset T of W with $|T| = l$ and $\sum_{w\in T} \pi(w) \geq d$, the set $U = \{(w), (w') \mid w \in T\}$ is valid for $(Q, D, k, F_{\mathsf{mono}}, B)$.

(2.2) Upper bound. When $\lambda = 1$, $\mathsf{RDC}(\mathrm{FO}, F_{\mathsf{mono}})$ is in #P, since it is in PTIME to check whether a set U is valid for $(Q, D, k, F_{\mathsf{mono}}, B)$ for fixed FO queries.

This completes the proof of Theorem 8.3 □

COROLLARY 8.4. *For a predefined constant k,*

— *the combined complexity bounds given in Theorems 5.1, 5.2, 6.1, 6.2, 7.1 and 7.2 are unchanged for* QRD, DRP *and* RDC*, respectively; and*
— *the data complexity is in*
  — PTIME *for* QRD*,*
  — PTIME *for* DRP*, and*
  — FP *for* RDC*,*

no matter whether for F_{MS}, F_{MM} or F_{mono}, and for CQ, UCQ, $\exists$FO$^+$or FO.

PROOF. We study QRD, DRP and RDC when k is a predefined constant, *i.e.,* we consider only candidate sets U with a constant size. We first establish their combined complexity, and then investigate their data complexity.

(1) Combined complexity. We first show the lower bounds, followed by upper bounds.

(1.1) Lower bound. Observe that lower bounds proofs of QRD$(\mathcal{L}_Q, F)$, DRP$(\mathcal{L}_Q, F)$ and RDC$(\mathcal{L}_Q, F)$ given for Theorem 5.1, 5.2, 6.1, 6.2, 7.1 and 7.2 are established by using $k = 2$ when F is F_{MS}, and by setting $k = 1$ when F is F_{MM} or F_{mono}, except those for QRD$(\mathcal{L}_Q, F)$ and DRP$(\mathcal{L}_Q, F)$, when F is F_{MS} or F_{MM} for CQ, UCQ and $\exists$FO$^+$. Hence these lower bounds hold here. Moreover, QRD$(\mathcal{L}_Q, F)$ and DRP$(\mathcal{L}_Q, F)$ are also shown to be NP-hard and coNP-hard in Theorem 8.2, respectively, for CQ, UCQ and $\exists$FO$^+$, by also using $k = 2$ when F is F_{MS}, and by setting $k = 1$ when F is F_{MM}. Hence these lower bounds hold here.

(1.2) Upper bound. The upper bounds of QRD$(\mathcal{L}_Q, F)$, DRP$(\mathcal{L}_Q, F)$ and RDC$(\mathcal{L}_Q, F)$ given for Theorem 5.1, 5.2, 6.1, 6.2, 7.1 and 7.2 obviously remain intact in the special case when k is a constant.

(2) Data complexity. We show that QRD(FO, F) and DRP(FO, F) are in PTIME, and RDC(FO, F) is in FP for fixed FO queries, when F is F_{MS}, F_{MM} or F_{MM}.

(2.1) QRD$(\mathit{FO},)$. We show that QRD(FO, F) is in PTIME by giving a PTIME algorithm. Given Q, D, δ_{rel}, δ_{dis}, F, k and B, the algorithm checks whether there exists a valid set U for (Q, D, k, F, B). It works as follows:

1. compute $Q(D)$;
2. enumerate all subsets U of $Q(D)$ such that $|U| = k$; denote by $\mathcal{S}$ the collection of all such sets U;
3. check whether there exists a set $U \in \mathcal{S}$ such that $F(U) \geq B$; if so, return "yes"; otherwise, return "no".

We show the algorithm is in PTIME when F is F_{MS}, F_{MM} or F_{mono}. Indeed, $Q(D)$ is PTIME computable since Q is fixed. Moreover, there are only polynomial many sets U in $\mathcal{S}$ that need to be checked in step 3 since k is a constant. Obviously, $F_{\mathsf{MS}}(U)$ and $F_{\mathsf{MM}}(U)$ are PTIME computable; F_{mono} is also PTIME computable since it is in PTIME to compute $Q(D)$. Thus step 3 can be done in PTIME. Hence QRD(FO, F) is in PTIME for fixed queries and constant k, when F is F_{MS}, F_{MM} or F_{mono}.

(2.2) DRP(FO, F). We show that DRP(FO, F) is in PTIME by giving a PTIME algorithm. Given Q, D, δ_{rel}, δ_{dis}, F, U, k and r, the algorithm works as follows:

1. compute $Q(D)$, in PTIME;
2. enumerate all subsets V of $Q(D)$ such that $|V| = k$; denote by $\mathcal{S}$ the collection of all such sets V; sort all sets in $\mathcal{S}$ in descending order based on their F values;
3. check whether $\mathsf{rank}(U) \leq r$; if so, return "yes"; otherwise return "no".

To see that the algorithm is in PTIME, note that it is in PTIME to compute $Q(D)$ since Q is fixed, and that there are only polynomial many sets in $\mathcal{S}$. Moreover, F is PTIME computable for F_{MS}, F_{MM} and F_{mono} when Q is fixed. Thus steps 2 and 3 are also in PTIME. In particular, step 3 can be easily done by counting the number of sets S in the sorted $\mathcal{S}$ that have $F(S) \geq F(U)$. Hence the algorithm is in PTIME.

(2.3) RDC($\textbf{\textit{FO}}, F$). We show that RDC(FO, F) is in FP by giving a PTIME algorithm. Given Q, D, δ_{rel}, δ_{dis}, F, k and B, the algorithm works as follows:

1. compute $Q(D)$, in PTIME;
2. enumerate all subsets U of $Q(D)$ such that $|U| = k$ and sort them in descending order based on their $F()$ values;
3. count and return the number of distinct valid sets for (Q, D, k, F, B).

Its step 1 is in PTIME since Q is fixed. Moreover, there are polynomial many subsets U in step 2 since k is a constant. Thus the algorithm is in PTIME, and the problem is in FP.

This completes the proof of Corollary 8.4. □

B. PROOFS OF SECTION 9

COROLLARY 9.2. *In the presence of compatibility constraints of $\mathcal{C}_m$, the combined complexity bounds of Theorems 5.1, 5.2, 6.1, 6.2, 7.1 and 7.2 remain unchanged for* QRD*,* DRP *and* RDC*.* □

PROOF. Observe that the lower bounds of QRD, DRP and RDC given in Theorems 5.1, 5.2, 6.1, 6.2, 7.1 and 7.2, respectively, are established when the compatibility constraints are absent. These bounds are obviously intact in the more setting when compatibility constraints are present.

Below we show that the upper bounds given there also remain intact in the presence of compatibility constraints Σ. To this end, we make minor changes to the algorithms given there by considering candidate sets for $(Q, D, \Sigma, k,)$. We show that the revised algorithms still work here and their complexity remain unchanged.

(1) QRD($\mathcal{L}_Q, F$). When F is F_{MS} or F_{MM}, we revise the algorithms given for Theorem 5.1 as follows: (a) the step 2 of the algorithm for QRD($\exists\text{FO}^+, F$) checks whether $|U| = k$, $F(U) \geq B$ and $U \models \Sigma$; (b) in step 2, the algorithm for QRD(FO, F) also checks whether $U \subseteq Q(D)$, $F(U) \geq B$ and $U \models \Sigma$. When F is F_{mono}, we revise the algorithm given in Theorem 5.2 such that in step 2, the algorithm checks whether $U \subseteq Q(D)$ and $U \models \Sigma$. Clearly, all the revised algorithms work here. Moreover, they are still in NP, PSPACE and PSPACE, respectively, since $U \models \Sigma$ can be checked in PTIME. Thus the upper bounds given in Theorems 5.1 and 5.2 carry over here.

(2) DRP($\mathcal{L}_Q, F$). When F is F_{MS} or F_{MM}, we revise the algorithms given for Theorem 6.1 such that (a) in step 2, the algorithm for DRP($\exists\text{FO}^+, F$) checks whether for each set $S \in \mathcal{S}$, $|S| = k$ and $S \models \Sigma$; (b) in step 2, the algorithm for DRP(FO, F) checks whether for each set $S \in \mathcal{S}$, $S \subseteq Q(D)$ and $S \models \Sigma$. When F is F_{mono}, the step 2 of the algorithm given for Theorem 6.2 checks whether for each $S \in \mathcal{S}$, $S \subseteq Q(D)$ and $S \models \Sigma$. Since we can check whether $S \models \Sigma$ is in PTIME, all the revised algorithms are still in coNP, PSPACE and PSPACE, respectively. Thus the upper bounds given in Theorem 6.1 and 6.2 remain valid here.

(3) RDC($\mathcal{L}_Q, F$). It suffices to show the following: (a) when F is F_{MS} or F_{MM}, it is in NP and in PSPACE to check if a set U is valid for (Q, D, Σ, k, F, B) for $\exists\text{FO}^+$queries Q and FO queries Q, respectively, and (b) when F is F_{mono}, it is in PSPACE to check whether

a set U is valid for (Q, D, Σ, k, F, B) for FO queries Q. Since it is in PTIME to check whether a set U satisfies Σ (*i.e.,* $U \models \Sigma$), both statements (a) and (b) hold here.

This completes the proof of Corollary 9.2. □

THEOREM 9.3. *In the presence of compatibility constraints of $\mathcal{C}_m$, the data complexity bounds of Theorems 5.4, 6.4 and 7.4 remain unchanged for* QRD, DRP *and* RDC, *respectively, for F_{MS} and F_{MM}. However, for F_{mono},*

— QRD *becomes* NP*-complete;*
— DRP *becomes* coNP*-complete; and*
— RDC *becomes* #P*-complete under parsimonious reductions.* □

PROOF. The lower bounds of $\mathsf{QRD}(\mathcal{L}_Q, F)$, $\mathsf{DRP}(\mathcal{L}_Q, F)$ and $\mathsf{RDC}(\mathcal{L}_Q, F)$ given in Theorems 5.4, 6.4 and 7.4, respectively, are established in the absence of compatibility constraints Σ, for F_{MS} and F_{MM}. These lower bounds obviously hold in the more general setting when Σ may be present. In addition, the upper bounds given there also carry over to the setting when compatibility constraints Σ are present. Indeed, along the same line as the proof of Corollary 9.2, we can revise the algorithms given in the upper bound proofs of Theorems 5.4, 6.4 and 7.4 by considering candidate sets for $(Q, D, \Sigma, k,)$. The revised algorithms still work in the presence of Σ with the same complexity, since one can check whether $U \models \Sigma$ in PTIME. Hence all the upper bounds still hold here.

We next show that for F_{mono}, the data complexity of QRD, DRP and RDC is NP-complete, coNP-complete and #P-complete under parsimonious reductions, respectively, in the presence of Σ, for CQ, UCQ, $\exists$FO$^+$and FO.

(1) $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{mono}})$. It suffices to show that $\mathsf{QRD}(\mathsf{CQ}, F_{\mathsf{mono}})$ is NP-hard and that $\mathsf{QRD}(\mathsf{FO}, F_{\mathsf{mono}})$ is in NP.

Lower bound. We verify that $\mathsf{QRD}(\mathsf{CQ}, F_{\mathsf{mono}})$ is NP-hard by reduction from 3SAT. Given an instance $\varphi = C_1 \wedge \ldots \wedge C_l$ of 3SAT defined over variables $X = \{x_1, \ldots, x_m\}$, we construct a database D, a fixed CQ query Q, functions δ_{rel}, δ_{dis} and F_{mono}, a set of fixed compatibility constraints Σ, a positive integer k and a real number B. We show that φ is satisfiable if and only if there exists a valid set U for $(Q, D, \Sigma, k, F_{\mathsf{mono}}, B)$.

(1) We use the database D given in the proof of Theorem 5.1 for $\mathsf{QRD}(\mathsf{CQ}, F_{\mathsf{MS}})$, over schema $R_C(\mathsf{cid}, L_1, V_1, L_2, V_2, L_3, V_3)$. That is, for each clause C_i, there are tuples in D to encode all truth assignments for variables in C_i that make C_i true.

(2) The query Q is an identity query on instances of R_C.

(3) For each tuple $t \in D$, we define $\delta_{\mathsf{rel}}(t, Q) = 1$, and for any other tuples t' of R_Q, we let $\delta_{\mathsf{rel}}(t, Q) = 0$. We define δ_{dis} as a constant function that returns 0 for each pair of tuples t and s of R_Q. We set $\lambda = 0$. Then for each set U of tuples of R_Q with k tuples, one can see that $F_{\mathsf{mono}}(U) = \sum_{t \in U} \delta_{\mathsf{rel}}(t, Q)$.

(4) We define Σ consisting of 10 constraints, given as follows:

$$\rho_1 : \forall t_1, t_2 : R_Q\big(t_1[\mathsf{cid}] = t_2[\mathsf{cid}] \rightarrow \bigwedge_{A \in R_C} t_1[A] = t_2[A]\big),$$

$$\rho_{ij} : \forall t_1, t_2 : R_Q\big(t_1[L_i] = t_1[L_j] \rightarrow t_1[V_i] = t_2[V_j]\big), i, j \in [1, 3].$$

Intuitively, constraint ρ_1 states that tuples in a set U have pairwise distinct cid-values, and the set $\{\rho_{ij} \mid i, j \in [1, 3]\}$ ensures that for each pair of tuples t and s in U, t and s agree on the values of their common variables. Note that all these constraints are defined on the fixed schema R_C (since Q is an identity query on R_c), and have the form of functional dependencies (see, *e.g.,* [Abiteboul et al. 1995] for functional dependencies). Intuitively, these compatibility constraints are used to assure that k-element sets U to be picked encode a valid truth assignment for variables X.

(5) Finally, we take $k = B = l$. That is, we only consider sets U that consist of l tuples, one for each clause in φ.

We next show that the formula φ is satisfiable if and only if there exists a valid set U for (Q, D, Σ, k, F, B), *i.e.,* the construction given above is indeed a reduction.

$\boxed{\Rightarrow}$ Assume first that φ is satisfiable. Then there exists a truth assignment μ_X^0 of X variables such that every clause C_j of φ is true under μ_X^0. Let U consist of l tuples of R_Q, one for each clause, in which the values for the variables in X agree with μ_X^0. Obviously, $U \models \Sigma$, and moreover, $F_{\mathsf{mono}}(U) = l \geq B$ by the definition of F_{mono}. Therefore, U is a valid set for (Q, D, Σ, k, F, B).

$\boxed{\Leftarrow}$ Conversely, assume that φ is not satisfiable. Suppose by contradiction that there exists a set $U \subseteq Q(D)$ such that $|U| = l$, $U \models \Sigma$, and $F_{\mathsf{mono}}(U) \geq B$. Then from the tuples in U, we can construct a valid truth assignment μ_X of variables in X that satisfies all clauses in φ, by the definition of δ_{rel}. This leads to a contradiction.

Upper bound. Observe that the algorithm for $\mathsf{QRD}(\mathrm{FO}, F_{\mathsf{mono}})$ revised in the proof of Corollary 9.2 works here. Clearly, its step 2 is in PTIME since one can check whether $U \models \Sigma$ in PTIME, $Q(D)$ is PTIME computable for a fixed FO query Q, and moreover, $F_{\mathsf{mono}}(U)$ is also in PTIME. Thus the algorithm is in NP when Q is fixed.

(2) $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{mono}})$. It suffices to show that $\mathsf{DRP}(\mathrm{CQ}, F_{\mathsf{mono}})$ is coNP-hard and that $\mathsf{DRP}(\mathrm{FO}, F_{\mathsf{mono}})$ is in coNP.

Lower bound. We verify the lower bound by reduction from the complement of 3SAT. Given an instance $\varphi = C_1 \wedge \ldots \wedge C_l$ of 3SAT over variables $X = \{x_1, \ldots, x_m\}$, we construct a database D, a fixed CQ query Q, functions δ_{rel}, δ_{dis} and F_{mono}, a fixed set Σ of compatibility constraints, a positive integer k, and a set U. We show that φ is not satisfiable if and only if $\mathsf{rank}(U) \leq r$. We use constant $r = 1$.

(1) We define the same formula $\varphi' = (\varphi \vee z) \wedge \bar{z} = \bigwedge_{i=1}^{l}(C_i \vee z) \wedge \bar{z}$, and the same database D (a single relation) over schema $R_C(L_1, V_1, L_2, V_2, L_3, V_3, Z, V_Z, A)$ as their counterparts given in the proof of Theorem 6.1. Here D consists of tuples that encode, for each clause $C_i \vee z$, all truth assignments μ_i for the three variables in C_i and z, and the truth value of the clause $C_i \vee z$ under μ_i. It also includes two extra tuples $(l+1, e_1, f_1, e_2, f_2, e_3, f_3, z, 1, 0)$ and $(l+1, e_1, f_1, e_2, f_2, e_3, f_3, z, 0, 1)$ for $\bar{z}$, where all e_i and f_i are distinct constants that are not in $X \cup \{z, 0, 1\}$.

(2) The query Q is an identity query on instances of R_c.

(3) Let U consist of $l+1$ tuples from D, one for each clause in φ' such that all variables in X and z are all set to be 1.

(4) Along the same line as the proof of $\mathsf{DRP}(\mathrm{CQ}, F_{\mathsf{mono}})$ given above, we define constraints Σ to ensure that for any set $U \subseteq Q(D)$, if $U \models \Sigma$, then the tuples in U have pairwise distinct cid-values and moreover, for each pair of tuples t and s in U, t and s agree on the values of their common variables.

(5) We define function δ_{rel} such that for each tuple $t \in D$, $\delta_{\mathsf{rel}}(t, Q) = 1$ if $t[A] = 1$; and for any other tuple t' of R_Q, we let $\delta_{\mathsf{rel}}(t', Q) = 0$. We set $\lambda = 0$. Then for each set U of tuples of schema R_Q, $F_{\mathsf{mono}}(U) = \sum_{t \in U}(\delta_{\mathsf{rel}}(t, Q))$.

(6) Finally, we set $k = l + 1$.

We next show that this is indeed a reduction.

$\boxed{\Rightarrow}$ Assume first that φ is satisfiable. Then there exists a truth assignment μ_X^0 for X variables that satisfies φ. We show that $\mathsf{rank}(U) = 2 > r = 1$. Let U^0 consist of $l+1$ tuples, one for each clause in φ', such that the values of all the variables in X agree with μ_X^0 and z is set to be 0. Obviously, for any tuple t in U^0, we have that $\delta_{\mathsf{rel}}(t, Q) = 1$ by the definition of δ_{rel}. Then $F_{\mathsf{mono}}(U^0) = l + 1$. Note that for each tuple $t \in U$, $t[A] = 1$ if $t \neq (l+1, e_1, f_1, e_2, f_2, e_3, f_3, z, 1, 0)$. Thus $F_{\mathsf{mono}}(U) = l$. Putting these together, we have that $\mathsf{rank}(U) \geq 2 > r = 1$.

$\boxed{\Leftarrow}$ Conversely, assume that φ is not satisfiable. Then there exists no truth assignment μ_X of X variables that satisfies φ. It is easy to see that for each candidate set S for (Q, D, k), there exist at most l tuples $t \in S$ such that $t[A] = 1$, and thus $F_{\mathsf{mono}}(S) \leq l = F_{\mathsf{mono}}(U)$. Therefore, $\mathsf{rank}(U) = 1 \leq r = 1$.

Upper bound. Clearly, the algorithm for $\mathsf{DRP}(\mathsf{FO}, F_{\mathsf{MS}})$ given in the proof of Corollary 9.2 also works here. Indeed, its step 2 is in PTIME since it is in PTIME to check whether $U \models \Sigma$, $Q(D)$ is PTIME computable for a fixed FO query Q, and moreover, $F_{\mathsf{mono}}(U)$ is also in PTIME. Thus the algorithm is in coNP here.

(3) $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$. It suffices to show that $\mathsf{RDC}(\mathsf{CQ}, F_{\mathsf{mono}})$ is #P-hard under parsimonious reductions, and that $\mathsf{RDC}(\mathsf{FO}, F_{\mathsf{mono}})$ is in #P.

Lower bound. We show that $\mathsf{RDC}(\mathsf{CQ}, F_{\mathsf{mono}})$ is #P-hard by parsimonious reduction from #SAT (see Section 7 for the details of #SAT). Given an instance $\varphi(X) = C_1 \wedge \cdots \wedge C_l$ of 3SAT over variables X, we construct the same $D, Q, \Sigma, \lambda, k, B$, and functions δ_{rel}, δ_{dis} and F_{mono} as their counterparts given above for $\mathsf{QRD}(\mathsf{CQ}, F_{\mathsf{mono}})$. We let $k = l$. It is easy to verify that μ_X is a truth assignment of X variables that satisfies $\varphi(X)$ if and only if there exists a valid set U for $(Q, D, \Sigma, k, F_{\mathsf{mono}}, B)$ encoding μ_X, *i.e.,* U consists of l tuples in D, one for each clause, in which the values for the variables in X agree with μ_X. Thus this is indeed a parsimonious reduction. Hence $\mathsf{RDC}(\mathsf{CQ}, F_{\mathsf{mono}})$ is #P-hard under parsimonious reductions.

Upper bound. We only need to show that it is in PTIME to check whether a set U is valid for $(Q, D, \Sigma, k, F_{\mathsf{mono}}, B)$ for a fixed FO query Q. Indeed, for a set $U \subseteq Q(D)$, it is in PTIME to check whether $|U| = k$, $U \models \Sigma$ and $F_{\mathsf{mono}}(U) \geq B$; in particular, $Q(D)$ is PTIME computable, and thus F_{mono} is PTIME computable. Hence $\mathsf{RDC}(\mathsf{FO}, F_{\mathsf{mono}})$ is in #P.

This completes the proof of Theorem 9.3. □

COROLLARY 9.4. *For identity queries, in the presence of compatibility constraints of $\mathcal{C}_m$, both the combined complexity and data complexity of Corollary 8.1 remain unchanged for* QRD, DRP, *and* RDC *for* F_{MS} *and* F_{MM}.

However, when it comes to F_{mono}*,*

— QRD *becomes* NP*-complete;*
— DRP *becomes* coNP*-complete; and*
— RDC *becomes* #P*-complete under parsimonious reductions,*

for both combined and data complexity. □

PROOF. In the presence of a set Σ of compatibility constraints, for identity queries we first study the combined and data complexity of QRD, DRP and RDC for F_{MS} and F_{MM}. We then investigate these problems for F_{mono}.

(1) When F is F_{MS} or F_{MM}. Observe the following. (a) We have shown in the proof of Corollary 8.1 that QRD, DRP and RDC are NP-hard, coNP-hard and #P-hard under parsimonious reductions, respectively, for fixed identity queries, when Σ is absent.

Thus the lower bounds carry over to the more general setting when Σ may be present. (b) Corollary 9.2 tells us that when Σ is present, QRD, DRP and RDC are in NP, coNP and #P, respectively, when Q is a CQ query that is not necessarily fixed. Observe that CQ subsumes identity queries. Hence for identity queries, QRD, DRP and RDC are in NP, coNP and #P, respectively, in the presence of Σ. Therefore, for identity queries, the combined complexity and data complexity of QRD, DRP and RDC are NP-complete, coNP-complete and #P-complete under parsimonious reductions, respectively, in the presence of Σ.

(2) When F is F_{mono}. Observe the following. (a) We have shown in the proof of Theorem 9.3 that QRD, DRP and RDC are NP-hard, coNP-hard and #P-hard under parsimonious reductions, respectively, for F_{mono}, by using a fixed identity query Q in the presence of Σ. Thus these lower bounds hold here. (b) The upper bounds given in the proof of Theorem 9.3 for QRD, DRP and RDC carry over here, since $Q(D)$ is PTIME computable when Q is an identity query, even if it is not fixed. From these we can see that for identity queries, the combined and data complexity of QRD, DRP and RDC are NP-complete, coNP-complete and #P-complete under parsimonious reductions, respectively, in the presence of Σ.

This completes the proof of Corollary 9.4. □

COROLLARY 9.5. *For λ = 0, in the presence of compatibility constraints of $\mathcal{C}_m$, the combined complexity bounds given in Theorem 8.2 remain unchanged for* QRD*,* DRP *and* RDC*, while the data complexity becomes*

— NP-*complete for* QRD*;*
— coNP-*complete for* DRP*; and*
— #P-*complete for* RDC *under parsimonious reductions,*

no matter for F_{MS}, F_{MM} and F_{mono}, and for CQ, UCQ, ∃FO⁺and FO. □

PROOF. For $\lambda = 0$, we first study the combined complexity of QRD, DRP and RDC, and then investigate their data complexity.

(1) Combined complexity. Observe the following. (a) The lower bounds given in Theorem 8.2 for QRD, DRP and RDC, respectively, are established when Σ is absent. Hence these lower bounds remain intact when Σ is possibly present. (b) The upper bounds given there carry over here when Σ is present. Indeed, along the same line as the proof of Corollary 9.2, we can revise the algorithms of Theorem 8.2 such that the revised algorithms work in the presence of Σ, with the same complexity since whether $U \models \Sigma$ can be checked in PTIME.

(2) Data complexity. We first study QRD, DRP and RDC for F_{MS} and F_{MM}, and then consider these problems for F_{mono}.

(2.1) When F is F_{MS} or F_{MM}. We show that in the presence of a set Σ of compatibility constraints, for $\lambda = 0$, QRD, DRP and RDC are NP-complete, coNP-complete and #P-complete under parsimonious reductions respectively, for fixed queries in CQ, UCQ, ∃FO⁺and FO.

(2.1.1) Lower bound. It suffices to verify that when F is F_{MS} or F_{MM}, $\mathsf{QRD}(\mathrm{CQ}, F)$, $\mathsf{DRP}(\mathrm{CQ}, F)$ and $\mathsf{RDC}(\mathrm{CQ}, F)$ are NP-hard, coNP-hard and #P-hard under parsimonious reductions, respectively, when $\lambda = 0$ and Σ is present.

We first show that $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MS}})$ and $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MM}})$ are NP-hard by reduction from 3SAT. Given an instance φ of 3SAT, we construct the same D, Q, Σ and functions δ_{rel} and δ_{dis} as their counterparts given in the proof of Theorem 9.3 for $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{mono}})$. Recall that δ_{rel} is defined as follows: $\delta_{\mathsf{rel}}(t, Q) = 1$ if $t \in D = Q(D)$, and for any other tuples t' of R_Q, $\delta_{\mathsf{rel}}(t', Q) = 0$. Thus, when $\lambda = 0$, for each set U consisting of k tuples in D, $F_{\mathsf{MS}}(U) = (k-1) \cdot \sum_{t \in U} \delta_{\mathsf{rel}}(t, Q) = (k-1) \cdot F_{\mathsf{mono}}(U)$, and $F_{\mathsf{MM}}(U) = \min_{t \in U} \delta_{\mathsf{rel}}(t, Q) = (1/k) \cdot F_{\mathsf{mono}}(U)$. Moreover, the lower bound of $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{mono}})$ in Theorem 9.3 is verified by setting $\lambda = 0$. Thus for $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MS}})$, we let $k = l$ and $B = (l-1) \cdot l$; and for $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MM}})$, we let $k = l$ and $B = 1$. Along the same line as the proof of Theorem 9.3, we can show that φ is satisfiable if and only if there exists a set U valid for (Q, D, Σ, k, F, B), when F is F_{MS} or F_{MM}.

We next verify that $\mathsf{DRP}(\mathrm{CQ}, F_{\mathsf{MS}})$ and $\mathsf{DRP}(\mathrm{CQ}, F_{\mathsf{MM}})$ are coNP-hard by reduction from the complement of 3SAT. Given an instance φ of 3SAT, we use the same reduction given in the proof of Theorem 9.3 for $\mathsf{DRP}(\mathrm{CQ}, F_{\mathsf{mono}})$, except the following: for each set U of k tuples of schema R_Q, (a) $F_{\mathsf{MS}}(U) = (k-1) \cdot \sum_{t \in U} \delta_{\mathsf{rel}}(t, Q)$; and (b) $F_{\mathsf{MM}}(U) = \min_{t \in U} \delta_{\mathsf{rel}}(t, Q)$. Recall that we show the lower bound of $\mathsf{DRP}(\mathrm{CQ}, F_{\mathsf{mono}})$ in the proof of Theorem 9.3 by using $\lambda = 0$. Thus along the same line as that proof, one can verify that φ is not satisfiable if and only if $\mathsf{rank}(U) \leq r$, for F_{MS} and F_{MM}.

Finally, we show that $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{MS}})$ and $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{MM}})$ are #P-hard by parsimonious reductions from #SAT. We have shown in the proof of Theorem 9.3 that $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{mono}})$ is #P-hard under parsimonious reductions for fixed queries, when $\lambda = 0$. Given an instance $\varphi(X)$ of #SAT over variables X, along the same line as the analysis in the proof of $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MS}})$ and $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{MM}})$ given above, we can use the same reduction given in the proof of Theorem 9.3 for $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{mono}})$, except the following: (a) for $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{MS}})$, we let $k = l$ and $B = (l-1) \cdot l$; and (b) for $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{MM}})$, we let $k = l$ and $B = 1$. Similarly, we can show that the number of truth assignments of the X variables that satisfies $\varphi(X)$ equals the number of valid sets for (Q, D, Σ, k, F, B), when F is F_{MS} or F_{MM}.

(2.1.2) Upper bound. We only need to show that $\mathsf{QRD}(\mathrm{FO}, F)$, $\mathsf{DRP}(\mathrm{FO}, F)$ and $\mathsf{RDC}(\mathrm{FO}, F)$ are in NP, coNP and #P, respectively, when F is F_{MS} or F_{MM}. Clearly, the upper bounds given in Theorem 9.3 for $\mathsf{QRD}(\mathcal{L}_Q, F)$, $\mathsf{DRP}(\mathcal{L}_Q, F)$ and $\mathsf{RDC}(\mathcal{L}_Q, F)$ for F_{MS} and F_{MM} carry over to the special case when $\lambda = 0$.

(2.2) When F is F_{mono}. Observe the following. (a) The lower bounds given in the Theorem 9.3 for the data complexity of $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{mono}})$, $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{mono}})$ and $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$ are established by using $\lambda = 0$. Thus the lower bounds hold here. (b) All the algorithms for these problems given there work in the special case when $\lambda = 0$. Thus the upper bounds carry over here. Hence when Σ is present, the data complexity of $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{mono}})$, $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{mono}})$ and $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$ are NP-complete, coNP-complete and #P-complete under parsimonious reductions, respectively.

This completes the proof of Corollary 9.5. □

COROLLARY 9.6. *For $\lambda = 1$, in the presence of compatibility constraints of $\mathcal{C}_m$, the combined complexity bounds given in Theorem 8.3 remain unchanged for* QRD, DRP *and* RDC.

The data complexity bounds of Theorem 8.3 remain unchanged for F_{MS} and F_{MM}. In contrast, for F_{mono} and for CQ, UCQ, $\exists$FO$^+$and FO,

— QRD *is* NP-*complete;*
— DRP *is* coNP-*complete; and*
— RDC *is* #P-*complete under parsimonious reductions.* □

PROOF. When $\lambda = 1$ and Σ is present, we study the combined complexity of QRD, DRP and RDC, followed by their data complexity.

(1) Combined complexity. Observe the following. The lower bounds given in Theorem 8.3 for QRD, DRP and RDC are established when Σ is absent. Thus the lower bounds hold in the more general setting when Σ is possibly present. Furthermore, the upper bounds given in Corollary 9.2 for QRD, DRP and RDC carry over here to the special case when $\lambda = 1$.

(2) Data complexity. We next study the data complexity for F_{MS}, F_{MM}, and F_{mono}.

(2.1) When F is F_{MS} or F_{MM}. In this setting, observe the following. (a) The lower bounds given in Theorem 8.3 for $\mathsf{QRD}(\mathcal{L}_Q, F)$, DRP and RDC are established when Σ is absent. Thus the lower bounds also hold in the presence of Σ. (b) The algorithms given in the proof of Theorem 9.3 for QRD, DRP and RDC work in the special cases when $\lambda = 1$. Therefore, the upper bounds given there remain valid the the special setting.

(2.2) When F is F_{mono}. We show that $\mathsf{QRD}(\mathcal{L}_Q, F_{\mathsf{mono}})$, $\mathsf{DRP}(\mathcal{L}_Q, F_{\mathsf{mono}})$ and $\mathsf{RDC}(\mathcal{L}_Q, F_{\mathsf{mono}})$ are NP-complete, coNP-complete and #P-complete under parsimonious reductions, respectively, for fixed queries in CQ, UCQ, $\exists\mathrm{FO}^+$and FO.

(2.2.1) Lower bound. We first study the lower bounds.

We show that $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{mono}})$ is NP-hard by reduction from 3SAT. Given an instance φ of 3SAT, we use the same reduction given in the proof of Theorem 9.3 for $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{mono}})$, except the following: (a) δ_{dis} is a constant function that returns 1 for any two different tuples of schema R_Q; (b) when $\lambda = 1$, for any set U of tuples of schema R_Q, $F_{\mathsf{mono}}(U) = \big(\lambda/(|Q(D)|-1)\big) \cdot \sum_{t \in U, t' \in Q(D)} (\delta_{\mathsf{dis}}(t, t'))$; and (c) $k = l$ and $B = l\cdot(l-1)/(|Q(D)|-1)$. Along the same line as the proof given there, one can readily verify that φ is satisfiable if and only if there exists a valid set U for $(Q, D, \Sigma, k, F_{\mathsf{mono}}, B)$.

We next show that $\mathsf{DRP}(\mathrm{CQ}, F_{\mathsf{mono}})$ is coNP-hard by reduction from the complement of 3SAT. Given an instance φ of 3SAT, We construct the same D, Q, U, Σ and r as their counterparts given in the proof of Theorem 9.3 for $\mathsf{DRP}(\mathrm{CQ}, F_{\mathsf{mono}})$, and moreover, we define the same function F_{mono} as the one given there when $\lambda = 1$. Along the same line as that proof, one can verify that φ is not satisfiable if and only if $\mathsf{rank}(U) \leq r = 1$.

We verify that $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{mono}})$ is #P-hard by parsimonious reduction from #SAT. Given an instance $\varphi(X)$ of #SAT, we construct the same D, Q, Σ, k, B and function F_{mono} as their counterparts given above for $\mathsf{QRD}(\mathrm{CQ}, F_{\mathsf{mono}})$. It is easy to verify that μ_X is a truth assignment of X variables that satisfies $\varphi(X)$ if and only if there exists a valid set U for $(Q, D, \Sigma, k, F_{\mathsf{mono}}, B)$ that encodes μ_X, such that U consists of l tuples in D, one for each clause, in which the values for the variables in X agree with μ_X. Thus it is indeed a parsimonious reduction. From this it follows that $\mathsf{RDC}(\mathrm{CQ}, F_{\mathsf{mono}})$ is #P-hard under parsimonious reductions.

(2.2.2) Upper bound. We have already shown in Theorem 9.3 that $\mathsf{QRD}(\mathrm{FO}, F_{\mathsf{mono}})$, $\mathsf{DRP}(\mathrm{FO}, F_{\mathsf{mono}})$ and $\mathsf{RDC}(\mathrm{FO}, F_{\mathsf{mono}})$ are in NP, coNP and #P, respectively, in the presence of Σ. Thus the upper bounds carry over here to the special case when $\lambda = 1$.

This completes the proof of Corollary 9.6. □

COROLLARY 9.7. *For a predefined constant k, in the presence of compatibility constraints of $\mathcal{C}_m$, the combined complexity and data complexity of Corollary 8.4 remain unchanged for* QRD, DRP *and* RDC, *respectively.* □

PROOF. We first study the combined complexity. Observe the following. (a) The lower bounds given in Corollary 8.4 for QRD, DRP and RDC are established for a (fixed) constant k when compatibility constraints are absent. Thus the lower bounds remain intact in the more general setting when compatibility constraints are present. (b) The upper bounds given there also carry over here. This follows from the fact that the algorithms developed in the proof of Corollary 9.2 for QRD, DRP and RDC in the presence of compatibility constraints obviously work in the special case when k is a constant.

For the data complexity, consider the algorithms given in the proof of Corollary 8.4 for $\mathsf{QRD}(\mathrm{FO}, F)$, $\mathsf{DRP}(\mathrm{FO}, F)$ and $\mathsf{RDC}(\mathrm{FO}, F)$. Along the same line as the proof of Corollary 9.2, we revise these algorithms to inspect candidate sets for (Q, D, Σ, k), by additionally checking whether $U \models \Sigma$. Clearly, the revised algorithms work here and the problems are still tractable, since it is in PTIME to check whether $U \models \Sigma$.

This completes the proof of Corollary 9.7. □