Measuring the Performance of Continuous Query Processing Approaches with *dief@t* and *dief@k*

Maribel Acosta[1] and Maria-Esther Vidal[2]

[1] Institute AIFB, Karlsruhe Institute of Technology
`maribel.acosta@kit.edu`
[2] Fraunhofer Institute, IAIS
`vidal@cs.uni-bonn.de`

Abstract. In this work, we present two experimental metrics named *dief@t* and *dief@k* which are able to capture and quantify the behavior of any system that produces results incrementally. We demonstrate the effectiveness of *dief@t* and *dief@k* on a generic SPARQL query engine able to produce results incrementally. Attendees will observe how both metrics are able to capture complementary information about the continuous behavior of the studied query engine. Moreover, valuable insights about the engine configurations that allow for continuously producing more answers over time will be observed. The demo is available at `http://km.aifb.kit.edu/services/dief-app/`.

1 Introduction

The continuous performance of approaches while executing queries is captured in answer traces. Answer traces record the points in time when a query engine produces an answer. To illustrate the concept of answer traces, consider that the SPARQL query presented in Figure 1(a) is executed with three variants of the nLDE engine [1]; the answer traces produced by the approaches when executing the query is depicted in Figure 1(b). In this case, the answer trace reveals that nLDE Not Adaptive exhibits a better performance than nLDE Random during the first 7.45 seconds of query execution. Therefore, answer traces provide further insights about the continuous efficiency – or diefficiency – of query engines.

In this work we analyze two metrics, *dief@t* and *dief@k* [2], able to capture the diefficiency during an elapsed time period t or while k answers are produced, respectively. In the demonstration, the attendees will be able to observe the effectiveness of both metrics to measure the performance of a SPARQL query engine able to produce results incrementally when executing a collection of queries. Moreover, in this work, we are providing the following resources:

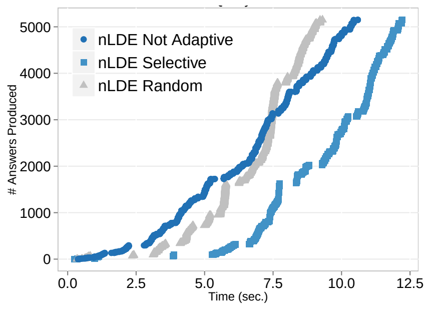
– The `dief` R package to compute the metrics *dief@t* and *dief@k*.[1]
– An online notebook that illustrates the usage of the `dief` package and reproduces the results reported by Acosta et al. [2].[2]
– An online demo to visualize the applications of *dief@t* and *dief@k*.[3]

[1] `https://github.com/maribelacosta/dief`
[2] `https://git.io/v7n8I`
[3] `http://km.aifb.kit.edu/services/dief-app/`

Retrieve information about resources classified as DBpedia alcohol and Yago alcohol.

```
SELECT * WHERE {
 ?d1 dc:subject dbc:Alcohols .
 ?d1 dbp:routesOfAdministration ?o .
 ?d1 dbp:smiles ?s .
 ?d2 a dbyago:Alcohols .
 ?d2 dbp:routesOfAdministration ?o .
 ?d2 dbp:molecularWeight ?w . }
```

(a) SPARQL Query



(b) Answer traces

Fig. 1. (a) SPARQL query executed in the running example and (b) answer traces. Answers produced (y-axis) in function of time (x-axis) when executing a query. The answer trace reveals that in the first 7.45 sec. of execution, nLDE Not Adaptive outperforms other approaches by producing more answers per time unit.

2 The Diefficiency Metrics: *dief@t* and *dief@k*

To measure the diefficiency of approaches, the metrics *dief@t* and *dief@k* [2] compute the area under the curve of answer traces. In this way, the *dief@t* and *dief@k* allow for capturing the continuous behavior of query processing engines, in contrast to other metrics that measure the performance of approaches at certain points in time, e.g., at the end of query execution.

The metric *dief@t* measures the diefficiency of an engine during the first t time units of query execution. Intuitively, approaches that produce answers at a higher rate in a certain period of time are more efficient.

***dief@t* interpretation**: Higher is better.

The metric *dief@k* measures the diefficiency of an engine while producing the first k answers when executing a query. Intuitively, approaches that produce a certain number of answers in a short period of time are considered more efficient.

***dief@k* interpretation**: Lower is better.

3 Demonstration of Use Cases

To illustrate the application of the *dief@t* and *dief@k* metrics, we recorded the answer traces of the nLDE query engine [1] when executing SPARQL queries with three different configurations: Not Adaptive, Random, and Selective. We use the nLDE Benchmark 1 [1] that comprises queries against the DBpedia dataset (v. 2015). The demo includes a total of 16 queries for which all the approaches produce more than one answer to compute the area under curve of the answer traces and to report on *dief@t* and *dief@k*. As a running example, consider the query Q9.sparql included in our online demo which corresponds to

Table 1. Query performance of the nLDE variants measured using *dief@t* at two different points in time: $\mathsf{t} = 9.3$ seconds and $\mathsf{t} = 4.5$ seconds.

(a) $\mathsf{t} = 9.3$

Approach	**Query**	***dief@t***
Selective	Q9.sparql	4,353.593
Random	Q9.sparql	12,975.313
Not Adaptive	Q9.sparql	14,090.062

(b) $\mathsf{t} = 4.5$

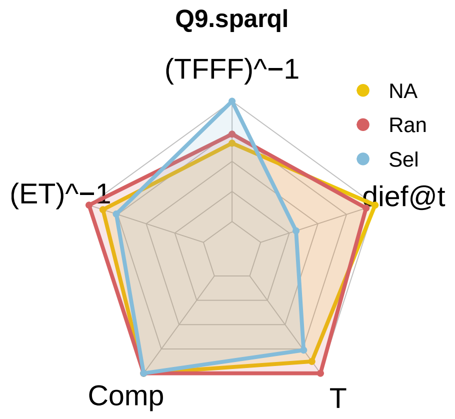
Approach	**Query**	***dief@t***
Selective	Q9.sparql	232.450
Random	Q9.sparql	648.694
Not Adaptive	Q9.sparql	1,590.766

the SPARQL query from Figure 1(a). After selecting a SPARQL query, the demo plots the answer trace (see Figure 1(b)) exhibited by the nLDE approaches.

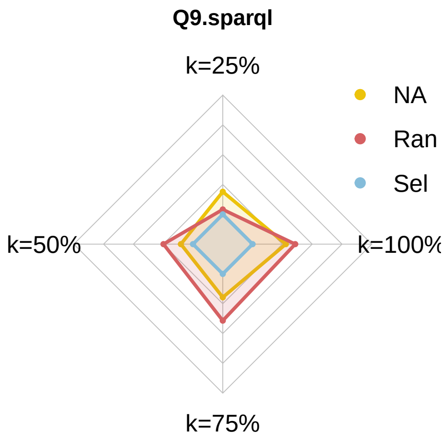
Measuring Performance with *dief@t*. This use case measures the performance of nLDE at different points in time. The application of *dief@t* allows for identifying engines that are able to produce incremental answers efficiently, which is particularly relevant in scenarios where engines are restricted to produce answers in a fixed time interval, e.g., before reaching a timeout. In this case, the demo allows for identifying approaches that exhibit the highest efficiency during the first *t* time units of query execution. Consider the query Q9.sparql and the point in time $\mathsf{t} = 9.3$. The nLDE variants Random and Not Adaptive achieved the highest *dief@t* values (see Table 1(a)); this indicates that nLDE Random and nLDE Not Adaptive exhibit a similar continuous performance during the first 9.3 seconds of execution. Furthermore, when $\mathsf{t} = 4.5$, the results of *dief@t* (see Table 1(b)) indicate that the nLDE Not Adaptive clearly outperforms the other approaches during the first 4.5 seconds of executing Q9.sparql.

Comparing *dief@t* with Other Metrics. This use case analyzes the performance of the nLDE variants using the metric *dief@t* as well as query processing metrics defined in the literature, including: execution time (**ET**), time for the first tuple (**TFFT**), answer completeness (**Comp**), and throughput (**T**). *dief@t* is computed at *t*, where *t* is the minimum execution time registered by one of the tested approaches when executing a query. To compare the performance of the studied approaches with multiple metrics, the demo presents the metric results as radar plots. For the sake of readability, the axes of the plot are transformed such that all the metrics have the same interpretation: higher is better. Figure 2(a) depicts the radar plot obtained for Q9.sparql. In Figure 2(a), the values of the metrics from the literature indicate that the three nLDE variants are competitive approaches. Nonetheless, *dief@t* suggests that nLDE Not Adaptive is able to continuously produce answers at a faster rate than the other approaches for this query until the fastest approach finalizes its execution. This indicates that *dief@t* allows for uncovering performance patterns of query engines that were not visible with metrics reported in the query processing literature.

Measuring *dief@k* at Different Answer Completeness. This use case measures the performance of nLDE while producing portions of the total answer. The application of *dief@k* allows for identifying engines able to produce the first answers more efficiently. This type of analysis is relevant in scenarios where users are interested in receiving only *k* answers or a portion of the total number



(a) Comparing *dief@t* with other metrics. Plot interpretation: Higher is better.



(b) *dief@k* at answer completeness k. Plot interpretation: Lower is better.

Fig. 2. Radar plots to compare the performance of nLDE with different metrics.

of answers. To compare the performance of the nLDE approaches at different answer completeness (25%, 50%, 75%, 100%), the demo presents the computed *dief@k* values in a radar plot. In this case, the interpretation of the plot is: lower is better. Figure 2(b) depicts the radar plot obtained for Q9.sparql. We observe that nLDE variants Random and Selective exhibit similar values of *dief@k* while producing the first 25% of the answers. However, when looking at *dief@k* at 100%, we can conclude that once nLDE Selective starts producing answers, it produces all the answers at a faster rate. This can be confirmed by inspecting the answer trace plot, where the trace for nLDE Selective has a higher slope over time in comparison with the other approaches.

4 Conclusions

The behavior of two novel experimental metrics, *dief@t* and *dief@k*, are demonstrated. Both measurement methods are able to capture and measure the continuous behavior of any computational system. As proof of concept, attendees observe the importance of capturing the continuous behavior of a SPARQL query engine. Patterns in the performance of this query engine are uncovered and discussed, as well as the usage of *dief@t* and *dief@k* for the evaluation of any other system that produces results incrementally.

References

1. M. Acosta and M. Vidal. Networks of linked data eddies: An adaptive web query processing engine for RDF data. In *ISWC*, pages 111–127, 2015.
2. M. Acosta, M. Vidal, and Y. Sure-Vetter. Diefficiency metrics: Measuring the continuous efficiency of query processing approaches. In *ISWC*, 2017.