Parallelised ABox Reasoning and Query Answering with Expressive Description Logics (Extended Abstract)[⋆]

Andreas Steigmiller and Birte Glimm

Ulm University, Ulm, Germany, <first name>.<last name>@uni-ulm.de

Handling knowledge bases that are formulated with expressive Description Logics (DLs), such as $\mathcal{SROIQ}$ [8], and that contain large amounts of facts is still challenging for state-of-the-art reasoning systems despite the huge range of developed optimisation techniques. For example, there are several summarisation [2,5] as well as abstraction techniques [6] and some problems are well-suited for a reduction to datalog [1,4,15]. However, these techniques do not necessarily work well for all ontologies, may be limited to certain queries or (fragments of) DLs, or require expensive computations (e.g., justifications). Particularly challenging is the support of conjunctive queries with complex concept terms and/or with existential variables that may bind to anonymous individuals since these features typically make it difficult to appropriately split the ABox upfront [13,14]. Although many tableau-based reasoning systems for expressive DLs directly integrate techniques that improve ABox reasoning, e.g., bulk processing with binary retrieval [7], caching and reusing the partial model (aka completion graph) from the initial consistency check [9,11], these techniques typically require significant amounts of main memory, which may be more than what is typically available.

We propose to dynamically split the model construction process with tableau algorithms. This allows for (i) handling larger ABoxes since not everything has to be processed at once and for (ii) exploiting parallelisation. In particular, we can ensure similarly sized work packages that can be processed concurrently without direct synchronisation. To ensure that the partial models constructed in parallel are "compatible" with each other, we employ a cache where selected consequences for individuals are stored and utilise appropriate reuse and expansion strategies in the model construction process for the cached consequences. Conjunctive query answering is supported by adapting the expansion criteria and by appropriately splitting the propagation work through the (partial) models.

For consistency checking, the work-flow with this *individual derivations cache* is roughly as follows: A thread gets a new part of the ABox assigned and retrieves stored derivations from the cache for the individuals in that part. The thread then tries to construct a fully expanded and clash-free *local* completion graph for the ABox part by reusing cached derivations and/or by expanding the

---

[⋆] The full technical report accompanying this extended abstract can be found at `https://www.uni-ulm.de/fileadmin/website_uni_ulm/iui.inst.090/Publikationen/2020/StGl2020-PARQA-TR-DL.pdf`.

Table 1: (Pre-)computation and accumulated query answering times for the evaluated ontologies with different numbers of threads in seconds (speedup factor in parentheses)

Ontology	(Pre-)computing				Query answering			
	K-1	K-2	K-4	K-8	K-1	K-2	K-4	K-8
ChEMBL	2170	1285 (1.7)	664 (3.3)	393 (5.5)	13108	8855 (1.5)	4659 (2.8)	3238 (4.0)
LUBM$_{800}$	2692	1385 (1.9)	828 (3.2)	439 (6.1)	2651	1762 (1.5)	995 (2.7)	559 (4.7)
Reactome	1340	701 (1.9)	419 (3.2)	363 (3.7)	883	387 (2.3)	303 (2.9)	222 (4.0)
Uniprot$_{100}$	1208	680 (1.8)	418 (2.9)	309 (4.0)	N/A	N/A	N/A	N/A
Uniprot$_{40}$	878	521 (1.7)	283 (3.1)	195 (4.4)	23	18 (1.2)	15 (1.5)	14 (1.6)
UOBM$_{500}$	1228	725 (1.7)	389 (3.2)	245 (5.0)	2565	1686 (1.5)	918 (2.8)	403 (6.4)

processing to individuals until they are “compatible” with the cache. Compatibility requires that the local completion graph is fully expanded and clash-free and that it can be expanded such that it matches the derivations for the remaining individuals in the cache. If it is required to extend the processing to some “neighbouring” individuals for achieving compatibility (e.g., if different non-deterministic decisions are required for the already processed individuals), then also the cached derivations for these individuals are retrieved and considered. If this process succeeds, the cache is updated with the new or changed derivations for the processed individuals. If compatibility cannot be obtained (e.g., due to expansion limitations that ensure similarly sized work packages), then the corresponding cache entries are marked such that these parts are considered later separately, i.e., a thread loads the data for (some) marked individuals and tries to construct a fully expanded and clash-free completion graph for the problematic part until full compatibility is obtained. If clashes occur that depend on reused (non-deterministic) derivations from the cache, then the corresponding individuals can be identified such that their expansion can be prioritized and/or the reuse of their derivations can be avoided. As a result, (in)consistency of the knowledge base can eventually be detected, as soon as all problematic individuals are directly expanded and all relevant non-deterministic decisions are investigated together. More details can be found in the accompanying technical report [10].

We implemented the proposed individual derivations cache with a few extensions and adaptations in the tableau-based reasoning system Konclude [12]. For evaluating the approach, we used the large ontologies and complex queries from the PAGOdA and VLog evaluations [3,15], which include the well-known LUBM and UOBM benchmarks as well as the real-world ontologies ChEMBL, Reactome, and Uniprot. We run the evaluations on a Dell PowerEdge R730 server with two Intel Xeon E5-2660V3 CPUs at 2.4 GHz and 512 GB RAM under a 64bit Ubuntu 18.04.3 LTS. Table 1 shows the times (and scalability) for the (pre-)computation phase (left-hand side), which excludes parsing and, hence, is dominated by consistency checking for these ontologies, as well as for answering the queries (right-hand side), accumulated for each ontology. K-1, K-2, K-4, K-8 stand for the versions of Konclude, where 1, 2, 4, and 8 threads are used, respectively. Without splitting the work with the individual derivations cache, the consistency checking runs out of memory for these ontologies and several queries cannot be computed. The parallelisation leads to significantly improved consistency checking and query answering times, but still leaves room for improvements for some ontologies and queries. As a comparison, PAGOdA reached the memory limit for one query and for two the used time limit of 10 hours.

Acknowledgements The work was funded by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) in project number 330492673.

References

1. Allocca, C., Calimeri, F., Civili, C., Costabile, R., Cuteri, B., Fiorentino, A., Fuscà, D., Germano, S., Laboccetta, G., Manna, M., Perri, S., Reale, K., Ricca, F., Veltri, P., Zangari, J.: Large-scale reasoning on expressive horn ontologies. In: Proc. 3rd Int. Workshop on the Resurgence of Datalog in Academia and Industry (Datalog 2.0'19). CEUR WS Proceedings, vol. 2368, pp. 10–21. CEUR (2019)
2. Bajraktari, L., Ortiz, M., Simkus, M.: Compiling model representations for querying large ABoxes in expressive DLs. In: Proc. 27nd Int. Joint Conf. on Artificial Intelligence (IJCAI'18). pp. 1691–1698 (2018)
3. Carral, D., Dragoste, I., González, L., Jacobs, C.J.H., Krötzsch, M., Urbani, J.: VLog: A rule engine for knowledge graphs. In: Proc. 18th Int. Semantic Web Conf. (ISWC'19). pp. 19–35. Springer (2019)
4. Carral, D., González, L., Koopmann, P.: From Horn-SRIQ to datalog: A data-independent transformation that preserves assertion entailment. In: Proc. 33rd AAAI Conf. on Artificial Intelligence (AAAI'19). pp. 2736–2743. AAAI Press (2019)
5. Dolby, J., Fokoue, A., Kalyanpur, A., Schonberg, E., Srinivas, K.: Scalable highly expressive reasoner (SHER). J. of Web Semantics **7**(4), 357–361 (2009)
6. Glimm, B., Kazakov, Y., Tran, T.: Ontology materialization by abstraction refinement in horn $\mathcal{SHOIF}$. In: Proc. 31st AAAI Conf. on Artificial Intelligence (AAAI'17). pp. 1114–1120. AAAI Press (2017)
7. Haarslev, V., Möller, R.: On the scalability of description logic instance retrieval. J. of Automated Reasoning **41**(2), 99–142 (2008)
8. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible $\mathcal{SROIQ}$. In: Proc. 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'06). pp. 57–67. AAAI Press (2006)
9. Sirin, E., Cuenca Grau, B., Parsia, B.: From wine to water: Optimizing description logic reasoning for nominals. In: Proc. 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'06). pp. 90–99. AAAI Press (2006)
10. Steigmiller, A., Glimm, B.: Parallelised abox reasoning and query answering with expressive description logics – technical report. Tech. rep., Ulm University, Ulm, Germany (2020), available online at `https://www.uni-ulm.de/fileadmin/website_uni_ulm/iui.inst.090/Publikationen/2020/StGl2020-PARQA-TR-DL.pdf`
11. Steigmiller, A., Glimm, B., Liebig, T.: Completion graph caching for expressive description logics. In: Proc. 28th Int. Workshop on Description Logics (DL'15) (2015)

12. Steigmiller, A., Liebig, T., Glimm, B.: Konclude: system description. J. of Web Semantics **27**(1) (2014)
13. Wandelt, S., Möller, R.: Distributed island-based query answering for expressive ontologies. In: Proc. 5th Int. Conf. on Advances in Grid and Pervasive Computing (GPC'10). pp. 461–470. Springer (2010)
14. Wandelt, S., Möller, R.: Towards ABox modularization of semi-expressive description logics. J. of Applied Ontology **7**(2), 133–167 (2012)
15. Zhou, Y., Cuenca Grau, B., Nenov, Y., Kaminski, M., Horrocks, I.: PAGOdA: Pay-as-you-go ontology query answering using a datalog reasoner. J. of Artificial Intelligence Research **54**, 309–367 (2015)