Parallelizing Algebraic Reasoning for the Description Logic $\mathcal{SHOQ}$

Jocelyne Faddoul
Centre for Logic and Information
St. Francis Xavier University
Nova Scotia, Canada
Email: jfaddoul@stfx.ca

Wendy MacCaull
Centre for Logic and Information
St. Francis Xavier University
Nova Scotia, Canada
Email: wmaccaul@stfx.ca

***Abstract*—Reaching the full potential of the semantic web awaits the availability of highly scalable reasoners. Despite numerous efforts to optimize existing Description Logics reasoners, there is always the need to compromise the expressivity or the size of the used ontologies in time sensitive applications. Hybrid algebraic reasoning has been investigated in the context of optimizing reasoning with ontologies where the expressivity is rich enough to include qualified cardinality restrictions and nominals. On the other hand parallel models have been considered to allow scalable reasoning with ontologies, however, only poor Description Logic expressivity has been considered. In this work, we investigate parallelizing hybrid algebraic reasoning as a means to seek scalable solutions without the need to sacrifice expressivity.**

I. Motivation

Applications of the semantic web are numerous, wide ranging and have tremendous potential for adding value in a vast array of situations which can take advantage of intelligence, i.e., the capacity to reason over knowledge stored in a knowledge base such as an ontology. However, if the application is time sensitive, the time required for reasoning can be prohibitive.

Description logics (DL) have gained a lot of attention in the research community as they provide a logical formalism for the codification of medical knowledge, ontologies, and the semantic web. There has been a great deal of research into optimizing DL reasoning strategies and in carving out fragments over which reasoning can proceed at a reasonable pace — but reasoning using these strategies or over these fragments often does not scale to allow the use of large ontologies. Reasoning for time sensitive tasks still requires severe restrictions on the expressivity, the complexity and/or the size of the ontology which, of course, limits the knowledge that can be used.

Standard DL inference services, e.g., TBox classification, concept satisfiability checking, instance checking, etc., have been extended with query answering in order to extract information and drive applications such as web services and workflow management systems [1]. For many applications (e.g., associated with health services delivery) these services are time sensitive, but require time consuming reasoning over complex and often large ontologies. The expressivity of the domain knowledge is often sacrificed in order to meet practical reasoning performance, hence the recent popularity of lightweight ontologies, i.e., expressed using the extensions of the tractable DL $\mathcal{EL}$. Sacrificing the expressivity of the knowledge modelled is a limiting (and often unacceptable) compromise. For example, given the Foundational Model of Anatomy (FMA) ontology[1], one might add axiom (1) to express the fact that the adult human has 206 bones, and axiom (2) to express the fact that the knee joint is involved in more than 100 rheumatic diseases[2]. These axioms use the qualified cardinality restrictions (QCRs) DL operator, which is known to lead to severe performance degradation of existing state of the art DL reasoners (e.g., Fact++[3], Hermit[4], Pellet[5]). RacerPro[6] remains the only DL reasoner that can efficiently handle QCRs using algebraic reasoning, however, it does not fully support nominals.

$$Adult \sqsubseteq Person \sqcap \geq 206\, hasBone \quad (1)$$

$$KneeJoint \sqsubseteq \geq 100\, involvedInDisease.RhDisease \quad (2)$$

To the best of our knowledge, algebraic reasoning remains the most promising approach for DL reasoning with ontologies relying on the use of QCRs. This has been shown in fragments of DL using Qualified Cardinality Restrictions (QCRs)[2], [3], inverse roles [4], [5], and nominals [6], [7]. Practical implementations of such algebraic tableau algorithms requires a carefully chosen set of optimizations in order to outperform the highly optimized existing state of the art reasoners. Most algebraic tableau-based algorithms proposed so far are double exponential in the worst case; their optimized implementations have been tested on a suite of artificial or often adapted subsets of ontologies. The scalability of the algebraic approach with real world and often large ontologies remains open.

The high performance computing (HPC) paradigm would seem to offer a solution to these problems, but progress using high performance computing methodologies has been challenging and slow [8], [9], [10]. The techniques that have offered speedy solutions in other domains (e.g., for "number

[1] http://sig.biostr.washington.edu/projects/fm/index.html
[2] http://www.medicinenet.com/knee_pain/article.htm
[3] http://owl.man.ac.uk/factplusplus/
[4] http://www.hermit-reasoner.com/
[5] http://clarkparsia.com/pellet/
[6] http://www.racer-systems.com/

crunching" in the physical sciences) do not suffice to crack the time bottleneck of reasoning tasks required for effective use of ontologies. Work is needed to find techniques for this kind of computing. The increasing availability of cloud computing facilities means that we can all have access to powerful computing resources; indeed, our own laptops have multiple cores. New methods are needed if we are to take advantage of their potential.

Recently, there has been encouraging results [11], [12], [10], [13], [14]. The work considered so far, considers parallelizing the TBox classification task [12], the Abox querying task [15], or the concept satisfiability checking task [8], [11] using ontologies relying on the least expressive fragments of DLs. Parallelizing algebraic reasoning to allow the handling of large ontologies using number restrictions needs further investigation.

Our research is focused on finding ways to combine high performance computing and algebraic tableau reasoning [6] to enable scalable reasoning with ontologies handling the expressivity of the DL $\mathcal{SHOQ}$.

II. High Performance Computing and Algebraic Reasoning

In this work, we investigate combining HPC and algebraic tableau reasoning for deciding DL concept satisfiability. Every standard DL reasoning task can be reduced to a concept satisfiability check. Our goal is to parallelize the algebraic tableau reasoning algorithm presented in [6] for the DL $\mathcal{SHOQ}$, which is basic DL $\mathcal{ALC}$ extended with transitive roles, role hierarchies, nominals and qualified cardinality restrictions, and for which the satisfiability problem is ExpTime-complete.

The algebraic algorithm presented in [6] decides the satisfiability of a concept C by constructing a compressed completion graph representing a model. The algorithm is hybrid; it relies on tableau expansion rules working together with an integer programming solver (e.g., simplex solver) and comes with a double exponential worst case complexity. However, in practice and when equipped with suited optimizations, algebraic reasoning performs better than existing state of the art reasoners in handling qualified cardinality restrictions and nominals [16], [7]. In this paper, we argue that algebraic reasoning is well suited for parallel programming models offering a potential improvement over standard tableau-based DL reasoning.

A. Parallel Reasoning

Constructing completion models for DL concepts often requires non-deterministic choices, which result in separately exploring more than one completion graph expansion. In the case of the DL $\mathcal{SHOQ}$, non-deterministic tableau-rules lead to an independent construction of tableau branches since nodes belonging to different branches do not exchange information. This feature suggests that we extend the search strategy adopted to construct tableau models using parallel processing. In the following, we list and compare the main sources of non-deterministic expansions in the cases of standard tableau DL reasoning and hybrid algebraic tableau DL reasoning for the DL $\mathcal{SHOQ}$.

a) Standard Tableau: In the case of the standard tableau algorithm for the DL $\mathcal{SHOQ}$ [17], non-determinism is due to:

- handling disjunctions (The $\sqcup$-Rule): if there exists in the completion graph a node x such that $C_1 \sqcup C_2$ is in the label, $\mathcal{L}(x)$, of x then there can be two possible and distinct ways to extend the completion model: one in which C_1 is added to the label of x, and one in which C_2 is added to the label of x.
- handling qualified cardinality restrictions ($choose$-rule, $\leq$-rule): if there exists in the completion graph a node x with $\leq nR.C$ its label and there exists $m >= 1$ nodes y, $y_1 \ldots y_m$, related to x via the role R, then:
  - for each y_m there can be two possible and distinct ways to extend the completion model: one in which C is added to the label of y_m, and one in which $\neg C$ is added to the label of y_m ($choose$-rule).
  - if $m > n$ there can be $\frac{m!}{n!}$ possible and distinct ways to extend the completion model such that in each case excess role fillers (y_i and y_j, $i \neq j$) are merged until the at-most restriction is satisfied ($\leq$-rule).

b) Algebraic Tableau: In the case of the hybrid algebraic tableau algorithm for the DL $\mathcal{SHOQ}$ [6], disjunctions are handled similar to the case of standard tableau. However, handling qualified cardinality restrictions relies on the use of the atomic decomposition technique [18], which computes disjoint partitions by considering all possible interactions between domain elements. This handling of domain elements results in only one additional source of non-determinism rather than the two sources for handling qualified cardinality restrictions with the standard tableau:

- handling partitions (the ch-Rule): for each partition computed by the atomic decomposition technique there can be two possible and distinct ways to extend the completion model: one in which the partition must be empty, and one in which the partition must have at-least one element.

We argue that hybrid algebraic reasoning appears to have a better potential for parallelization than standard tableau for the following reasons:

- having less sources of non-determinism (2 instead of 3) means less overhead in managing concurrent execution of non-deterministic rules. This also means that adopting optimizations such as dependency directed backtracking becomes more fine grained and less complicated.
- the ch-rule always fires for two choices. This means that the search trees resulting from the distinct branches have similar structure which facilitates load balancing between parallel expansions of the search tree. Load balancing is a common goal in parallel computing where unequal thread workloads can easily diminish the performance gain of parallelization.
- satisfying qualified cardinality restriction is delegated to an inequation solver and can be done in isolation from tableau expansion. This means that the task of satisfying

the inequations can be delegated to the use of separate threads, or even FPGAs [14] and GPUs (Graphical Processing Units).
- the use of "compressed completion graph" consisting of proxy nodes representing sets of domain elements instead of completion graphs consisting of a node representing each domain element allows the use of a smaller data structure representing the completion model. This means that a smaller amount of data needs to shared among and communicated between parallel tasks thus reducing the communication overhead between threads.

B. The Parallel Execution Framework

We consider parallelization of the hybrid algebraic reasoning algorithm using an object-oriented framework supporting thread level parallelism (TLP). In this framework, a compressed completion graph data structure is shared among threads which concurrently apply tableau rules until termination of the satisfiability check. In this approach we choose to investigate the or-parallelism with a shared memory strategy, where non-deterministic branches of the ch-Rule and the $\sqcup$-Rule are explored using parallel threads.

In order to minimize the overhead of creating and destroying threads every time a non-deterministic rule is applied, we implement the *Thread Pool* design pattern. This means that a fixed number of threads is created and organized into a queue until associated with an applicable completion rule. The number of threads can be assigned depending on the number of available processors and resource thrashing can be avoided. In this framework, threads coordinate themselves using the *Leader/Followers* design pattern where a single thread from the thread pool acts as a leader and manages thread-rule assignment. Figure 1 illustrates the state transitions between threads when adopting the Leader/Followers design pattern. When a thread is in the leader state, it can immediately change state to become in executing state if a non-deterministic completion rule becomes applicable. A thread in the executing state can run concurrently with the leader thread and other executing threads. Once a thread finishes expanding a completion graph, it either changes state to become leader, if no leader thread is available, or to become follower. In the latter case, a thread is waiting, in the thread pool, to be promoted to the leader state by the current leader. Since the threads expand a shared model, the compressed completion graph can be implemented as a *Monitor Object* to ensure synchronization between threads.

Even though the order in which expansion rules are applied does not affect soundness and completeness of the satisfiability test, in practice, results have shown that performance speedup can be achieved using certain ordering. We plan to investigate our parallel model while considering different ways of enforcing an ordering in which node labels are chosen as premise for tableau rules as was done in [11] for the basic DL $\mathcal{ALC}$.

III. Discussion

The implementation and evaluation of this parallel framework is ongoing work. The HARD (Hybrid Algebraic Reasoner for Description Logics) prototype reasoner [16], implemented in java, is being redesigned to adopt the parallel execution framework described in the previous section. Given that HARD has been equipped with a suite of crucial optimization techniques such as lazy partitioning and dependency directed backtracking, one has to consider the possible effects of the TLP. One of the features that renders non-deterministic rules appealing for parallelization, is that they result in completion graph expansions which can be explored in isolation. However, dependency directed backtracking relies on information exchange between branches such that dependencies must be recorded and consulted before pruning the search space. Such required communication between branches complicates the use of dependency directed backtracking in our parallel framework. In this context, the use of the *Thread-Specific Storage* or the *Monitor Object* seems worth investigating.

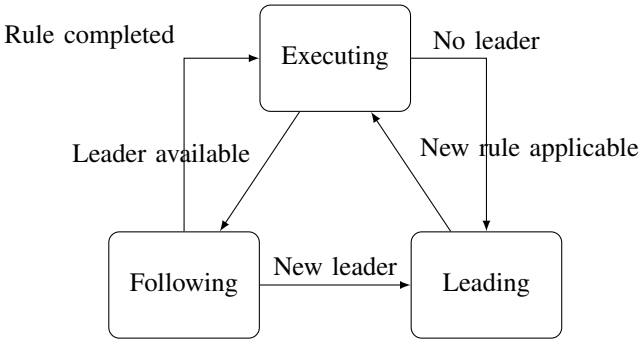

Fig. 1. Thread transitions in the Leader/Followers design pattern.

IV. Related Work and Outlook

Our work here is motivated by problems arising in the area of health services delivery. The healthcare system is composed of many different professionals operating at many sites of care offering a wide variety of services and requiring a vast amount of information both in the form of data and also in the form of clinical and other protocols. We are currently involved in a multi-year project in collaboration with our local health authority and industry partner to develop an ontology-driven Careflow Management System. Our lab has developed an ontology-driven workflow system and we have done some work in the scalability problem for querying over the OWL 2RL fragment [19], [20], [21] over large ABoxes. We are currently expanding our system with an ontology-driven service discovery engine. We believe that the high performance computing paradigm offers a great deal of hope for the scalability problem for knowledge crunching, that is, for ontological reasoning tasks, in time sensitive applications.

A parallel algorithm for description logics reasoning has been considered in 1995 [22], with limited scalability results due to hardware limitations. Further results and research activity have been reported in this area since the work presented in [8], where non-deterministic choices in core satisfiability test were explored concurrently. Parallelizing rule-based OWL inferencing has been considered in [13] by examining a data

partitioning approach and a rule partitioning approach. Parallel reasoning has also been investigated in the context of distributed resolution reasoning [23] about interlinked ontologies as an alternative to centralized tableau-based reasoning (DL $\mathcal{ALCHIQ}$). Techniques using the MapReduce algorithm to classify $\mathcal{EL}^+$ ontologies [24] and fuzzy $\mathcal{EL}^+$ ontologies [25] have been proposed with no empirical evaluation. Concurrent classification of lightweight ontologies has also been considered in the context of consequence-based reasoning [26]. Tableau-based concurrent classification of more expressive ontologies has been recently reported in [9], where lock-free algorithms with limited synchronization have been used in a multi-core environment, and in [10] where specialized data structures have been proposed to optimize the use of a shared memory environment.

We plan to investigate parallel reasoning in the context of enhancing core satisfiability tests for expressive ontologies. Little work has been reported in this context. In [11], a parallel search engine (Mozart system) was used to parallelize Description Logics satisfiability check, however, the algorithm only considers basic DL $\mathcal{ALC}$. We plan to handle the expressivity of the DL $\mathcal{SHOQ}$ by designing a parallel architecture for the algebraic tableau calculus presented in [6], and which was shown to be the only one able to decide the satisfiability of complex ontologies relying on the use of nominals and qualified cardinality restrictions [16], [7]. We plan to implement and evaluate our approach in a multi-core and multi-processor environment using the Atlantic Computational Excellence Network (ACEnet) resources.

ACKNOWLEDGMENT

The first author is an ACEnet Postdoctoral Fellow, and gratefully acknowledges support from Atlantic Canada's HPC Consortium; the second author acknowledges support from NSERC and ACOA.

REFERENCES

[1] W. MacCaull and F. Rabbi, "NOVA Workflow: A Workflow Management Tool Targeting Health Service Delivery," in *International Smposium on Foundations of Health Information Engineering and Systems (FHIES - 2011)*, ser. Lecture Notes in Computer Science, vol. 7151. Springer, 2012, pp. 75–92.

[2] V. Haarslev, M. Timmann, and R. Möller, "Combining tableaux and algebraic methods for reasoning with qualified number restrictions," in *Proceedings of the International Workshop on Description Logics (DL'2001), Aug. 1-3, Stanford, USA*, ser. CEUR Workshop Proceedings, vol. 49, 2001, pp. 152–161. [Online]. Available: citeseer.ist.psu.edu/article/haarslev01combining.html

[3] N. Farsiniamarj and V. Haarslev, "Practical reasoning with qualified number restrictions: A hybrid abox calculus for the description logic," *AI Communications*, vol. 23, no. 2-3, pp. 205–240, 2010.

[4] Y. Ding, "Tableau-based reasoning for description logics with inverse roles and number restrictions," Ph.D. dissertation, Concordia University, 2008.

[5] L. R. Pour, "Algebraïc reasoning with the description logic $\mathcal{SHIQ}$," Master's thesis, Concordia University.

[6] J. Faddoul and V. Haarslev, "Algebraic tableau reasoning for the description logic $\mathcal{SHOQ}$," *Journal of Applied Logic*, vol. 8, no. 4, pp. 334–355, 2010.

[7] ——, "Optimized algebraic tableau reasoning for the description logic $\mathcal{SHOQ}$," *Journal of Artificial Intelligence Research (JAIR) - In preparation*, 2013.

[8] T. Liebig and F. Müller, "Parallelizing tableaux-based description logic reasoning," in *Proceedings of the 2007 workshop on On the Move to Meaningful Internet Systems 2007 - OTM 2007*, 2007, pp. 1135–1144.

[9] M. Aslani and V. Haarslev, "Concurrent classification of owl ontologies - an empirical evaluation," in *Proceedings of the 2012 International Workshop on Description Logics, DL-2012*, Y. Kazakov, D. Lembo, and F. Wolter, Eds., vol. 846, Rome, Italy, June 7-10, 2012.

[10] K. Wu and V. Haarslev, "A parallel reasoner for the description logic $\mathcal{ALC}$," in *Proceedings of the 2012 International Workshop on Description Logics, DL-2012*, ser. CEUR Workshop Proceedings, Y. Kazakov, D. Lembo, and F. Wolter, Eds., vol. 846. Rome, Italy, June 7-10: CEUR-WS.org, 2012.

[11] A. Meissner, "Experimental analysis of some computation rules in a simple parallel reasoning system for the $\mathcal{ALC}$ description logic," *International Journal of Applied Mathematics and Computer Science*, vol. 21, no. 1, pp. 83–95, March 2011.

[12] M. Aslani and V. Haarslev, "Parallel tbox classification in description logics - first experimental results," in *ECAI 2010 - 19th European Conference on Artificial Intelligence*, ser. Frontiers in Artificial Intelligence and Applications, H. Coelho, R. Studer, and M. Wooldridge, Eds., vol. 215. Lisbon, Portugal, August 16-20: IOS Press, 2010, pp. 485–490.

[13] R. Soma and V. Prasanna, "Parallel inferencing for owl knowledge bases," in *37th International Conference on Parallel Processing - ICPP'08*, 2008, pp. 75–82.

[14] P. Subramanian, "A field programmable gate array based finite-domain constraint solver," Master's thesis, School of Graduate Studies, Utah State University, 2008.

[15] J. E. M. Alvarez, "Query engine for massive distributed ontologies using mapreduce," Master's thesis, Technische Universitat Hamburg-Harburg, 2010.

[16] J. Faddoul, "Reasoning algebraïcally with description logics," Ph.D. dissertation, Concordia University, Montreal, Canada, 2011.

[17] I. Horrocks and U. Sattler, "Ontology reasoning in the $\mathcal{SHOQ}$(D) description logic," in *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001)*. Morgan Kaufmann, Los Altos, 2001, pp. 199–204. [Online]. Available: download/2001/ijcai01.pdf

[18] H. J. Ohlbach and J. Koehler, "Modal logics description logics and arithmetic reasoning," *Artificial Intelligence*, vol. 109, no. 1-2, pp. 1–31, 1999.

[19] F. Rabbi and W. MacCaull, "T-square: A domain specific language for rapid workflow development," in *ACM/IEEE 15th International Conference on Model Driven Engineering Languages & Systems (MODELS 2012)*, ser. Lecture Notes in Computer Science, vol. 7590, Innsbruck, Austria, September 2012, pp. 36–52.

[20] M. R. U. Faruqui and W. MacCaull, "Owlontdb: A scalable reasoning system for OWL 2 RL ontologie," in *FHIES 2012: International Symposium on Foundations of Health Information Engineering and Systems*, ser. Lecture Notes in Computer Science, vol. 7789. Springer, 2013.

[21] F. Rabbi, W. MacCaull, and M. R. U. Faruqui, "A scalable ontology reasoner via incremental materialization," in *Submitted to CBMS 2013*.

[22] F. W. Bergmann and J. J. Quantz, "Parallelizing description logics," in *19th Ann. German Conference on Artificial Intelligence*, ser. LNCS. Springer-Verlag, 1995, pp. 137–148.

[23] A. Schlicht and H. Stuckenschmidt, "Distributed resolution for expressive ontology networks," in *Web Reasoning and Rule Systems, 3rd International Conference (RR-2009)*, Chantilly, VA, USA, October 2009, pp. 87–101.

[24] R. Mutharaju, F. Maier, and P. Hitzler, "A MapReduce algorithm for el+," in *23rd International Workshop on Description Logics*, 2010, pp. 464–474.

[25] Z. Zhou, G. Qi, C. Liu, P. Hitzler, and R. Mutharaju, "Reasoning with fuzzy-$\mathcal{EL}^+$ ontologies using mapreduce," in *ECAI 2012 - 21st European Conference on Artificial Intelligence*, L. D. R. et al., Ed. IOS Press, 2012, pp. 933–934.

[26] Y. Kazakov, M. Krötzsch, and F. Simancík, "Concurrent classification of el ontologies," in *Proceedings of the 10th international conference on The semantic web*, ser. ISWC' 11. Bonn, Germany: Springer-Verlag, 2011, pp. 305–320.