Automata-Based Quantitative Reasoning

Suguman Bansal

Abstract. Existing solution approaches for problems in *quantitative analysis* suffer from two challenges that adversely impact their theoretical understanding, and large-scale applicability due to limitations on scalability. These are the *lack of generalizability*, and *separation-of-techniques*. Lack of generalizability refers to the issue that solution approaches are often specialized to the underlying *cost model* that evaluates the quantitative property. Different cost models deploy such disparate algorithms that there is no transfer of knowledge from one cost model to another. Separation-of-techniques refers to the inherent dichotomy in solving problems in quantitative analysis. Most algorithms comprise of two phases: A *structural phase*, which reasons about the structure of the quantitative system(s) using techniques from automata or graphs; and a *numerical phase*, which reasons about the quantitative dimension/cost model using numerical methods. The techniques used in both phases are so unlike each other that they are difficult to combine, forcing the phases to be performed sequentially, thereby impacting scalability.

This abstract summarizes my thesis work [2], which contributes towards a novel framework that addresses the aforementioned challenges. The introduced framework, called *comparator automata* or *comparators* in short, builds on automata-theoretic foundations to generalize across a variety of cost models. The crux of comparators is that they enable automata-based methods in the numerical phase, hence eradicating the dependence on numerical methods. In doing so, comparators are able to integrate the structural and numerical phases. On the theoretical front, we demonstrate that comparator-based solutions have the advantage of generalizable results, and yield complexity-theoretic improvements over a range of problems in quantitative analysis. On the practical front, we demonstrate through empirical analysis that comparator-based solutions render more efficient, scalable, and robust performance, and hold the ability to integrate quantitative with qualitative objectives.

Many classic questions in formal methods can be seen as involving comparisons between different system runs or inputs. For instance, the classical *model checking problem* of verifying if a system S satisfies a linear-time temporal specification P [11]. Traditionally, this problem is phrased language-theoretically: S and P are interpreted as sets of (infinite) words, and S is determined to satisfy P if $S \subseteq P$. The problem, however, can also be framed in terms of a *comparison* between words in S and P. Suppose a word w is assigned a weight of 1 if it belongs to the language of the system or property, and 0 otherwise. Then determining if $S \subseteq P$ amounts to checking whether the weight of every word in S is less than or equal to its weight in P [1]. The ubiquity of comparisons becomes more pronounced in quantitative analysis: Firstly because every system execution is assigned a real-valued cost. W.l.o.g, we can assume that the cost model is an *aggregate function* $f : Z^\omega \to \Re$. The cost of an execution is related to the aggregate function f applied to the weight-sequence corresponding to the execution; Secondly, because problems in quantitative analysis reduce to comparing the cost of executions to a constant value (such as in quantitative games), or more generally to the cost of another execution (as in quantitative inclusion).

Keeping comparisons at the center, we introduce a language-theoretic/automata-theoretic formulation of the comparison between weighted sequences. We introduce *comparator automata* (*comparators*, in short), a class of automata that read pairs of infinite weight sequences synchronously and compare their aggregate values in an online manner [5]. Formally, a *comparator automata for aggregate function f, relation* R*, and upper bound* $\mu > 0$ is an automaton that accepts a pair $(A, B) \in (\Sigma \times \Sigma)^\omega$ of sequences of bounded integers, where $\Sigma = \{-\mu, \mu-1, \ldots, \mu\}$, iff $f(A)\ \mathsf{R}\ f(B)$, where $\mathsf{R} \in \{>, <, \geq, \leq, \neq =\}$ is an inequality or equality relation. We say a comparator is *ω-regular* if it is finite-state and accepts by the Büchi condition. We show that for several aggregation functions such as limsup and liminf, discounted-sum (DS) with integer discount factors, ω-regular aggregate functions [10], the comparator automata is a Büchi automaton. While for several other functions, such as limit-average and DS with non-integer discount factors, the comparator cannot be ω-regular.

The first benefit of comparators is that one can design *generalizable solutions* to a variety of problems in quantitative analysis for all aggregate functions that permit an ω-regular

comparator. This is in contrast to existing approaches where solutions and results are very specific to the cost-models. For instance, consider the problem of quantitative inclusion for aggregate function f, called *f-inclusion* in short. From prior work, even complexity-theoretic results are diverse. While quantiative inclusion is PSPACE-hard, there is ample variance in upper bounds. Quantitative inclusion is PSPACE-complete under limsup/liminf [9] and EXPTIME for DS with integer discount factors [8,9]. Instead, with comparators, we can generalize to show that for all aggregate functions with an ω-regular comparators, quantitative inclusion is PSPACE-complete. Thus these generalization solutions can also sometimes even result in tighter complexity-theoretic bounds (eg. DS inclusion for integer discount-factors). Similarly, we design generalize solutions for solving quantitative games with perfect and imperfect information. What enables these generalizable results is the fact that since Büchi automata are closed under all set-theoretic operations, ω-regular comparators can easily be operated on in the generic algorithms that we design.

Another benefit of comparators has to do with the separation-of-techniques challenge of quantitative analysis. *Separation-of-techniques* refers to the inherent dichotomy in existing algorithms for problems in quantitative analysis. Most algorithms comprise two phases. In the first phase, called the *structural analysis phase*, the algorithm reasons about the structure of the quantitative system(s) using techniques from automata or graphs. In the second phase, called the *numerical analysis phase*, the algorithm reasons about the quantitative dimension/-cost model using numerical methods. For eg. to solving DS inclusion with integer discount factors consists of two phases: a Determination phase where a non-deterministic system is determinized (the structural phase) followed by linear programming (numerical phase). The techniques used in both these phases are so unlike each other that they cannot be solved *in tandem*. Hence the phases have to be performed sequentially, hence affecting their scalability.

The advantage of our automata-based formulation of comparators in this aspect is that comparators reduce the numerical problem of comparison of aggregation of weight sequences into one of membership in an automaton. This way, enabling both phases, the structural phase, and numerical phase, of quantitative analysis to be performed using automata-based techniques. Subsequently, creating an opportunity to design *integrated methods* as opposed to separation-of-techniques methods for problems in quantitative analysis. The generalizable algorithms that we design when the aggregate function permits ω-regular comparators are based on reducing the problems in quantitative analysis to those in qualitative analysis. In particular, quantitative inclusion is reduced to *language inclusion*, and quantitative games under perfect and imperfect information are both reduced to solving *parity games*.

Apart from the theoretical development of the comparator-framework, my thesis also investigates the practical considerations of comparator-based algorithms. For this, we investigate problems with DS aggregation. We first observe that DS comparators have an even more specialized structure that being ω-regular - They are safety or co-safety automata [7]. We can leverage this additional property to design theoretically and practically superior algorithms fro DS inclusion with integer discount factors [4,7], and quantitative games with DS with perfect information [3]. We are also able to design the first pragmatic *anytime algorithm* for DS inclusion with a non-integer discount factor, a problem for which even decidability is currently open [6].

In summary, the progress of comparator-based approaches has bridged quantitative reasoning to qualitative reasoning. The use of language-theoretic properties such as safety/co-safety was unheard of in the context of quantitative analysis. But now with comparators, we have opened these avenues. While we have demonstrated the benefits of comparators on a few problems, we believe its reach can cast a wide web, and encourage us to re-think algorithms in quantitative analysis.

References

1. Christel Baier, Joost-Pieter Katoen, et al. *Principles of model checking*. MIT press Cambridge, 2008.
2. Suguman Bansal. Automata-based quantitative reasoning. Doctoral thesis (Submitted), Rice University, 2020.
3. Suguman Bansal, Krishnendu Chatterjee, and Moshe Y. Vardi. On the analysis of quantitative games. In *preparation*.
4. Suguman Bansal, Swarat Chaudhuri, and Moshe Y. Vardi. Automata vs linear-programming discounted-sum inclusion. In *Proc. of CAV*, 2018.
5. Suguman Bansal, Swarat Chaudhuri, and Moshe Y. Vardi. Comparator automata in quantitative verification. In *Proc. of FOSSACS*, 2018.
6. Suguman Bansal and Moshe Y. Vardi. Anytime discounted-sum inclusion. In *submission*.
7. Suguman Bansal and Moshe Y. Vardi. Safety and co-safety comparator automata for discounted-sum inclusion. In *Proc. of CAV*, 2019.
8. U. Boker and T. A. Henzinger. Exact and approximate determinization of discounted-sum automata. *LMCS*, 10(1), 2014.
9. K. Chatterjee, L. Doyen, and T. A. Henzinger. Quantitative languages. *Transactions on Computational Logic*, 11(4):23, 2010.
10. Swarat Chaudhuri, Sriram Sankaranarayanan, and Moshe Y. Vardi. Regular real analysis. In *Proc. of LICS*, pages 509–518, 2013.
11. Edmund M. Clarke, E Allen Emerson, and A Prasad Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 8(2):244–263, 1986.