MEDINFO 2015: eHealth-enabled Health
I.N. Sarkar et al. (Eds.)

doi:10.3233/978-1-61499-564-7-947

A Scalable Architecture for Rule Engine Based Clinical Decision Support Systems

Soumi Chattopadhyay[1], Ansuman Banerjee[1], Nilanjan Banerjee[2]

[1]Indian Statistical Institute, Kolkata, India
[2]IBM Research, India

Abstract

Clinical Decision Support systems (CDSS) have reached a fair level of sophistication and have emerged as the popular system of choice for their aid in clinical decision making. These decision support systems are based on rule engines navigate through a repertoire of clinical rules and multitudes of facts to assist a clinical expert to decide on the set of actuations in response to a medical situation. In this paper, we present the design of a scalable architecture for a rule engine based clinical decision system.

Keywords:

Clinical decision support, Rule engine

Introduction

The core component of CDSS [1] consists of a rule inferencing system where characteristics of an individual patient are matched to a clinical knowledge base, and then patient-specific assessments or recommendations are presented to the clinician(s) and/or the patient for a decision. Commercial rule engines such as Drools [2] and ILOG [3] have shown to satisfy most needs of a CDSS. An evaluation of the performance of ILOG showed that the overall performance was generally satisfactory for small rule sets. However, hundreds of rule sets are possible in a real-time surveillence system and rule engines do not perform very well on large rule sets. In this paper, we propose an efficient and scalable framework for a rule engine based CDSS.

The Proposed Framework

The key idea behind the framework is a cache-based lazy loading mechanism that consists of rule clustering and hashing kernel, coupled with a prediction based technique for rule evaluation and faster actuation. The main motivation for rule clustering is to save the memory requirement for rule loading and to expedite rule firing by the rule prediction strategy. If we can efficiently and accurately predict the set of rules that are expected to be evaluated to truth based on the arriving facts, we can improve the rule firing time by triggering the actuations beforehand. The rule clustering step is done during preprocessing, whereas prediction is done at runtime. In the clustering step, two rules that have a common fact variable in their antecedents are put in the same cluster. The clusters are mutually disjoint. This ensures that only the rules in one cluster are affected by a fact and only that cluster gets loaded into production memory based on the matching facts at once. The prediction mechanism at run time only stores the last evaluation outcome of a rule in history. In this work, we experiment with two different prediction schemes: a deterministic scheme, in which a rule R is predicted to be true if it was true in the previous session, and a probabilistic scheme, in which the prediction is done with a certain probability.

Experimental Results

We implemented the framework with *JRuleEngine* [4] and evaluated its performance with a dataset of 16 rules defined over 32 fact variables. The rule base contains 5 clusters. Performance evaluation experiments were done using randomly generated sets of facts. Table 1 shows the memory gain (in %), the amount of memory saved using lazy loading of rules and used more intelligently for larger rule bases, with lazy loading of rules.

Table 1. No. of facts vs memory gain (in %)

Cardinality of Facts	*1*	*5*	*10*	*15*	*30*
Memory Gain(%)	*87.5*	*18.75*	*5.56*	*3.94*	*1.125*

Figure 1 shows the actuation time required in normal mode of operation, i.e., without prediction and with different prediction algorithms discussed above, against the number of facts. Our prediction based approaches for rule evaluation gains consistently over the normal mode of operation.

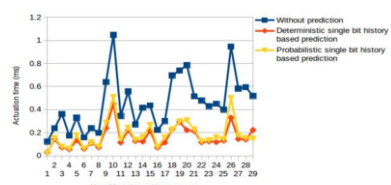

Figure 1. Number of facts vs actuation time

Conclusion

Despite all the benefits of rule engine based CDSS, they suffer from some performance bottlenecks. We propose a framework for improving the overall performance of the rule engine in terms of memory utilization and execution time. Experimental results show promising performance gain that can be leveraged for large-scale operations. In future, we plan to take up the evaluation of our proposed architecture for industry grade performance benchmarks.

References

[1] Berlin et al., A taxonomic description of computer-based clinical decision support systems, Journal of Biomedical Informatics, 2006.

[2] DROOLS: http://www.drools.org/

[3] ILOG: http://www-01.ibm.com/software/info/ilog/

[4] JRULEENGINE: http://jruleengine.sourceforge.net/