

# Z-log: Applying System-Z

Michael Minock and Hansi Kraus  
*mjm@cs.umu.se c97hks@cs.umu.se*

The University of Umeå, Sweden

**Abstract.** We present Z-log – a practical system that employs the system-Z[14] semantics. Z-log incurs polynomial cost for compilation and entailment in the horn and q-horn[3] cases. Z-log’s complexity is intractable in the unrestricted case – but intractable in the number of defaults that cause the violation of the q-horn property.

We present here initial performance results over two alternative rule-bases. The results indicate that Z-log currently scales to problems on the order of 1000’s of propositional rules when the rules are in q-Horn form. We shall be applying Z-log in cognitive disease diagnosis.

## 1 Introduction

Efforts have revolved around extending KLM[10] equivalent semantics to more practical *adventurous* semantics [14][6][9][4][11]. Although these semantics exhibit many desirable features of common sense, in general they have rather high computational cost. For example, in the general case, entailment in system-Z[7] requires  $O(\log(n))$  satisfiability tests, once  $O(n^2)$  satisfiability tests are issued to compile a knowledge base of  $n$ -defaults. And system-Z is the computationally least expensive semantics of the aforementioned and the only one that is tractable in the horn and q-horn cases[5]. The q-horn[3] case represents the combination of disguised horn clauses<sup>1</sup> and 2-SAT problems. Q-horn form thus extends common horn form, allowing for limited use of disjunction in the antecedents of rules.

During compilation system-Z builds a unique ranked partition of the rule-base. The kappa value of a propositional formula ( $\phi$ ) is the highest rank at which the formula and the rules at such a rank and higher are inconsistent. The kappa value indicates the degree of surprise associated with  $\phi$  - that is  $P(\phi) < \epsilon^{\kappa(\phi)}$ . This  $\kappa$  function is the basis of the system-Z entailment semantics[14] [7].

Although there have been prototype implementations of system-Z[8][4], and an exact counterpart to system-Z using possibility measures[2], Z-log represents the most advanced direct system-Z implementation to date. For example DRS[4] is a semantically based system, calculating an actual rank for all possible worlds. As such it is only applicable to systems of approximately 16 propositions and 20 defaults. To be fair the intention of DRS was to explore the virtues of Maximum Entropy with variable strength rules, not to build a practical system-Z ‘system’.

---

<sup>1</sup> That is CNFs where propositions may be renamed to reduce the number of positive literals to at most 1 per clause.

## 2 Z-log

We are calling our implementation of the system-Z<sup>2</sup> Z-log. See [13] for an in-depth discussion of Z-log's optimization techniques and its (optional) extended semantics  $\hat{Z}$ . The key point is that the system is tractable in q-horn cases. Moreover the degree of intractability of the system depends on the number of rules that cause the violation of the q-horn property. Thus when the number of such non-q-horn rules is bounded, we may consider their cost constant and may consider problems that contain larger numbers of 'friendly' q-horn form rules.

Z-log is run by providing a definition file (.def) in which strong and default propositional rules are specified and a query file (.qry) where facts are asserted (or retracted) and a stream of entailment and  $\kappa$  queries are processed. The following is the definition file `catAndMouse.def`.<sup>3</sup>

```
/* We may (optionally) declare propositions before we use them with '+'. */
+cheese cheddar tasty dirty . // Features of the cheese,
+starving seekCheese. // the mouse,
+catAtHome catInKitchen catInDen, catInLivingRoom catInBedroom.// the cat
+night. // and the world.

/* The defaults: prolog style, order unimportant */
cheese := cheddar. // A hard rule.
tasty :- cheese. // A default.
!tasty :- cheese & dirty.
tasty :- cheese & dirty & starving.

!catInKitchen := catInDen.
!catInLivingRoom := catInDen.
!catInLivingRoom := catInKitchen.
!catInBedroom := true. // false is also a built-in proposition.

catInKitchen | catInLivingRoom | catInDen :- catAtHome.
catInDen :- night.
seekCheese :- !catInKitchen & cheese.
seekCheese :- starving & cheese.
!seekCheese :- catInKitchen.
```

The following is a query file that corresponds to a fact context and a stream of queries over the definition file `sylvester.qry`.

```
cheddar.
!catInLivingRoom.
?tasty. // TRUE.
?catInKitchen. // UNDECIDED.
?!catInBedRoom. // TRUE.
?seekCheese. // UNDECIDED.
night.
?catInKitchen. // FALSE.
?seekCheese. // TRUE.
dirty.
?tasty. // FALSE.
starving.
?tasty. // TRUE.
```

---

<sup>2</sup> Our implementation also supports  $\hat{Z}$ .  $\hat{Z}$  enables the inheritance of stable properties to exceptional subclasses.

<sup>3</sup> Note the PROLOG like syntax. The default  $\neg a \wedge b \rightarrow c$  is written `c :- !a & b`. The hard rule  $a \Rightarrow b \vee c$  is written `b | c := a`. Comments are as in C.

```

/* Kappa value queries */
#cheese & !tasty. // is 1.
#night & !catInDen. // is 1 under Z, 2 under Z-hat.

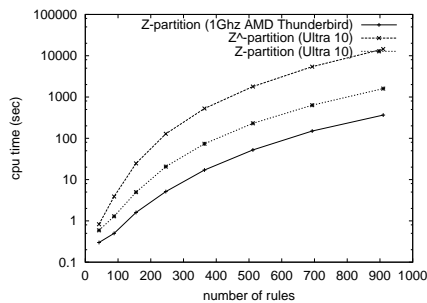
```

At the command line one types:

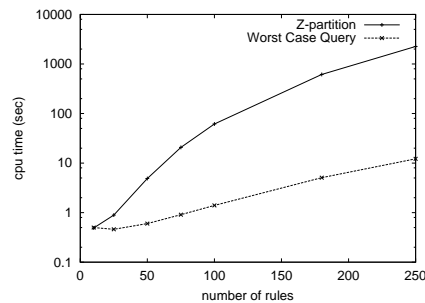
```
java Zlog -Z^ catAndMouse.def sylvester.qry
```

This sends to standard output the results from compiling `catAndMouse.def` and then processing all the queries in `sylvester.qry` under  $\hat{Z}$  semantics. Note that a `catAndMouse.par` file is written to disk so Z-log does not need to recompute the partition over multiple program invocations.

## 2.1 Performance



(Figure 1)



(Figure 2)

Figure 1 shows the costs for partition construction in a simplified, non-deterministic version of Wumpus world. It is non-deterministic in that forward actions may or may not be performed, and simplified in that the world is an array of black and white tiles. Although this is, no doubt, a toy problem, such a spatial-temporal reasoning problem generates many hundreds of propositional rules for even small array sizes. The CPU times for entailment queries over this Wumpus world example uniformly take less than 10 seconds.

Figure 2 shows an example generating the worst case partition depth for system Z. In this example the rule  $p_1 \rightarrow q$  is followed by  $p_1 \wedge p_2 \rightarrow \neg q$  which is followed by  $p_1 \wedge p_2 \wedge p_3 \rightarrow q$ , and so forth. The query is, given  $p_1$ , does  $q$  follow?

## 3 Medical Decision Support - Cognitive Disorder Diagnosis

We believe that Z-log may be of utility within a medical decision support tool. *Medical decision support systems* give advice to medical personal. Such advice is relative to a patient's complex and often incomplete medical record. Published

clinical guidelines serve as a knowledge source from which to author initial defaults and hard constraints. The knowledge-base is then refined by medical personnel to capture unwritten knowledge of the domain.

We are currently building a Z-log knowledge-base devoted to the problem of computing cognitive disorder diagnosis. Our knowledge was obtained from clinical guidelines and discussions with physicians[12]. Currently we have a system with approximately 100 hard rules and 30 defaults. The advice that the system generates seems sound. Soon we shall solicit feedback from medical personal.

## 4 Conclusion

We have described the first implementation of Z-log: a q-horn tractable system that encodes system-Z[14] and  $\hat{Z}$ [13]. In the future we intend to more precisely characterize the knowledge bases and types of problems over which Z-log may be practically applied. In addition we shall run the system under a wider class of benchmarks.

## References

1. S. Benferhat, C. Cayrol, D. Dubois, J. Lang, and H. Prade. Inconsistency management and prioritized syntax-based entailment, 1993.
2. Salem Benferhat, Didier Dubois, and Henri Prade. Representing default rules in possibilistic logic. *KR 1992: 673-684*, 1992.
3. E. Boros, Y. Crama, and P. Hammer. Polynomial-time inference of all valid implications for horn and related formulae. *Annals of Mathematics and Artificial Intelligence*, 1:21–32, 1990.
4. R. Bourne and S. Parsons. Maximum entropy and variable strength defaults, 1999.
5. Thomas Eiter and Thomas Lukasiewicz. Complexity results for default reasoning from conditional knowledge bases. In *KR2000: Principles of Knowledge Representation and Reasoning*, pages 62–73, San Francisco, 2000. Morgan Kaufmann.
6. H. Gefner. Default reasoning: causal and conditional theories, 1992.
7. Goldszmidt and Judea Pearl. Qualitative probabilities for default reasoning, belief revision, and causal modeling. *Artificial Intelligence*, 84(1-2):57–112, 1996.
8. M. Goldszmidt. *Qualitative Probabilities: A Normative Framework for Commonsense Reasoning*. PhD thesis, University of California, California, 1992.
9. Moisés Goldszmidt, Paul Morris, and Judea Pearl. A maximum entropy approach to nonmonotonic reasoning. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, Menlo Park, CA, 1990. AAAI Press.
10. S. Kraus, D. Lehmann, and M. Magidor. Preferential models and cumulative logics. *Artificial Intelligence*, 44:167–207., 1990.
11. Daniel J. Lehmann. Another perspective on default reasoning. *Annals of Mathematics and Artificial Intelligence*, 15(1):61–82, 1995.
12. Helena Lindgren. Decision support systems and diagnosing cognitive disorders. Master's thesis, University of Umeå, Sweden, 2000.
13. Michael Minock and Hansi Krause. Z-log: A system-z (and  $\hat{Z}$ ) 'system'. Technical Report 02.05, The Univeristy of Umea, Umea, Sweden, May 2002.
14. J. Pearl. A natural ordering of defaults with tractable applications to default reasoning. *Proceedings of the 3rd Conference on Theoretical Aspects of Reasoning about Knowledge*, pages 121–135., 1990.