Pacific Knowledge Systems

An Introduction to RippleDown

An introduction to RippleDown Technology, the problems it addresses, and how it relates to other knowledge system technologies

Paul Compton
University of New South Wales
2011

Contents

The Problem

It is reasonably straightforward to build a simple knowledge base, and such knowledge bases can often outperform humans in the reliability of their decisions since they never overlook any of the data they have been taught to reason about.

It is quite a different task to build a knowledge base that emulates all the subtleties of human decision-making.

Not appreciating this difference is the main reason expert systems have, in general, not lived up to their early promise of emulating the subtlety of human expertise. The challenge is that the knowledge base needs to know about all the different contexts that may arise if it is to deal with these contexts in the same way as a human expert. The obstacle to addressing this challenge is that, although a human expert can deal with each concrete context that arises and justify their conclusions in this context, this is a very different task from enumerating and elucidating all possible contexts that may occur. The philosophical literature on situated cognition argues that it is impossible for experts to elucidate all contexts out of context, but whether or not it is impossible, it is clearly extremely difficult.

So, any knowledge-base built with the help of a human expert will need to be amended when these inevitable new contexts arise that it does not yet know about. The alternative approach is to try and avoid reliance on the human expert with some variant of machine learning or case-based reasoning to build a system.

But then the same sorts of problems occur in a different guise:

- The training data set now needs to cover every context
- The training data needs to be accurately classified in all the subtlety that the system will be required to emulate in the future
- For machine learning, each context needs to be presented in sufficient volume so that knowledge can be abstracted from data.

So whilst machine learning systems have been extremely successful and valuable, their success is limited to the more coarse-grained classifications (analogous to “simple” Knowledge Bases).

The RippleDown Solution

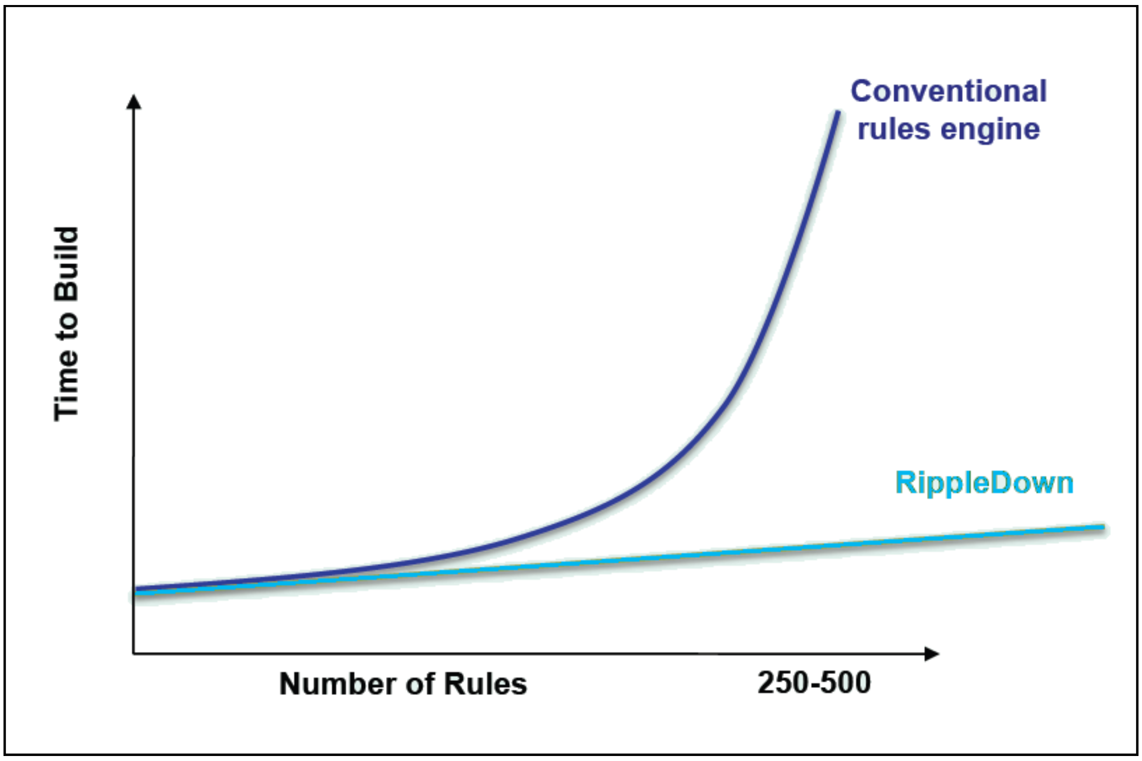
Pacific Knowledge Systems' proprietary and patented RippleDown technology is based on an area of research known as Ripple-Down Rules. A web search finds over 200 papers covering a wide range of active research related to Ripple-Down Rules.

The Ripple-Down Rules approach aims at making it sufficiently simple to add knowledge to deal with new contexts as and when they arise, so that the whole system can be built at the same time it is being used.

No longer is there any need for an extended Knowledge Base development and testing period before a system can go "live". This goal has been achieved by Pacific Knowledge Systems with its RippleDown product suite, with customers building thousands of rules at an average rate of between one and two minutes per rule, including verification that the change is an incremental improvement to the knowledge base.

This rate of rule addition is effectively independent of knowledge base size – whether the knowledge base has 10 or 10,000 rules.

This speed of knowledge acquisition makes it realistic to have the knowledge base in use while new knowledge is added to deal with new contexts as they arise. Pacific Knowledge Systems' customers have over 150 knowledge bases in routine use, all built and maintained in this way.



RippleDown Technology

Current business rule literature and earlier expert system literature emphasise the importance of declarative knowledge and the distinction between declarative knowledge and the inference engine that processes it. That is, it has been generally assumed that knowledge building is a quite separate process to knowledge delivery (inferencing). However, there is, in fact. a complex and hidden interaction between the declarative knowledge and the inference engine. The declarative knowledge does not directly indicate the order in which the knowledge should be processed. This order is determined by the conflict resolution strategies of the inference engine. That is, what is the outcome if two rules apply but each have conflicting conclusions? Which rule should take precedence? It turns out that the conflict resolution must depend on features of the knowledge, so in turn, there is implicit control of the inferencing by the knowledge itself.

Hence as the knowledge base becomes more complex a change in the knowledge can cause all sorts of unexpected and unintended side-effects resulting in different conclusions from inferencing. Each change to the Knowledge Base therefore requires more and more testing as the Knowledge Base gets larger, in order to deal with these inevitable side-effects.

The consequence is that, anecdotally, no more than about five rules per day can be added to a large (traditional) knowledge base. Furthermore, large knowledge bases eventually become unmaintainable as organisational knowledge about how the knowledge base is structured is lost with the passing of time or staff moving on.
The result of all this is that whilst small Knowledge Bases can be built relatively easily, the large Knowledge Bases needed for complex domains require a prohibitively large and on-going effort of an expensive resource (i.e. a human expert) to develop and maintain. How does RippleDown solve this problem?

Firstly, by ensuring inference is completely determined and controlled by the addition of knowledge. Each rule has explicit links to the rules be evaluated next, depending on whether the rule is satisfied on not. A new rule is automatically linked into the structure at the point of “failure” – whether the mistake is due to an error or omission or some context not yet considered. This locating is automatic, not something that needs to be considered or “designed in” by an expert.

Secondly, if a rule has made a mistake, a new rule is added to the structure as a correction to the original rule. With the addition of this new rule, the case is now dealt with correctly. A “case” is simply the set of data to be processed by the Knowledge Base in a single inferencing transaction.

All rules are added in a RippleDown® Knowledge Base in this way. Each case requiring a new rule to be added is linked to that rule, and is called a “cornerstone” case as it provides the foundation or context for that new rule.
When adding a new rule, the RippleDown® structure determines which of other, past cornerstone cases might also reach the new rule. These are precisely the cases which might have their interpretation changed by the new rule.

When adding a new rule, the RippleDown structure determines which of other, past cornerstone cases might also reach the new rule. These are precisely the cases which might have their interpretation changed by the new rule.

If necessary, the human expert will be asked to identify features in the new case which distinguish it from past cornerstone cases. This ensures the past cornerstone cases will still be processed the same way with the same conclusion. Alternatively the expert might decide that the new conclusion is more appropriate for the cornerstone cases.

This review step is included in the knowledge acquisition process, which, even with over 10,000 rules, takes only a minute or two per rule added. A useful side-effect of this process is that it assists a human expert to be consistent across different contexts.

Rather than asking an expert to think of all possible contexts up front – an impossible task - the system may point out past cases to which the rule applies but where the expert previously reached a different conclusion. Did the expert make a mistake previously? Did they overstate the case? Have they changed their mind now? Or are there some other features that distinguish this case that the expert did not point out previously?

This has great value in new domains, such as genetic testing, where expertise is still evolving, in helping ensure that the expertise that evolves is at least consistent. Most RippleDown users will use the approach to build a knowledge base from the ground up, but the approach can be used equally well as a wrapper around an existing system to maintain and extend it.

RippleDown can also be integrated with machine learning if, for some parts of the knowledge domain, sufficient cases are available. One of the obvious advantages of the RippleDown approach is that the system can be started and put into use immediately without a long start-up phase, accumulating cases and building and testing the knowledge base.

On the other hand, RippleDown seems to have an extra cost, in the requirement that the output should to be monitored by the human expert to ensure that cases are picked up for which rules should be added – i.e. to detect the inevitable "mistakes" that an immature Knowledge Base will make.

In fact this is not an 'extra' requirement as any system where the output is of critical importance needs to be monitored, unless the system is guaranteed to be completely correct. However, in practice, correctness can never be guaranteed for complex knowledge domains because invariably all possible cases have not been tested.
So a requirement for long-term monitoring of a RippleDown system is actually the same as for other systems – it is a function of criticality of decisions in the domain, not the technology itself.

The difference with the RippleDown process is that the role of monitoring is explicit. In fact, the monitoring process becomes an integral part of the Knowledge Acquisition process – new rules are added to the live system if and when a mistake is detected during monitoring. As the system develops the expert can set whatever level of monitoring they consider is appropriate for different conclusions and contexts, given the criticality of the conclusions and the rate of amendments that have been required to date.

Some conclusions will be always monitored, while at the other extreme, others may never need to be monitored after a phase of initial development. The decision on the level of monitoring needed becomes an explicit part of the whole process of using a knowledge-based system.

The experience of Pacific Knowledge Systems' customers is that in a high transaction environment, over 90% of the interpretations of a mature Knowledge Base will not need monitoring and can be sent out automatically. The RippleDown system retrieves and presents the 10% to be monitored as part of the process.

Application Domains

The Ripple Down Rule approach can be applied to virtually any knowledge problem and one can find papers developing research systems for areas as diverse as tuning genetic algorithms, identifying people's actions in video surveillance, text processing and web monitoring. Pacific Knowledge Systems customers currently use RippleDown technology in over 80 expert domains.

A small sample is as follows:

- Providing detailed patient-specific interpretations for laboratory reports based on all available patient data and history
- Continuous monitoring of the test results for patients in a critical care unit (e.g. cardiac unit) to detect alert conditions and so notify the emergency response team
- Controlling workflow processes within a laboratory, for example, the decision whether or not a slide of a certain blood sample needs to be made
- Checking whether clinicians' diagnostic requests are appropriate given the previous tests ordered for the patient, standard guidelines and insurance rules
- Checking in real-time whether all the details of a patient's test registration is correct, thus avoiding clinical or billing problems down the track
- Checking re-issued airline tickets to ensure that travel agents have applied an airline's fare rules correctly
- Composing letters to clients based on information collected from various sources.

Business Advantages

The obvious advantage of RippleDown technology is that very large knowledge systems can actually be built and deployed. However, there are also some key advantages in how this is done:

- The system is built while it is in use, so experts don't have to be taken out of their normal duties
- Knowledge acquisition is a very small addition to an expert's other duties, with the trade-off there is an almost immediate saving as the system starts to relieve the expert of those tasks now performed by the Knowledge Base, freeing them up for more interesting and challenging tasks
- The domain expert is responsible for development and maintenance of the Knowledge Base, not the IT department. So the experts have a high level of ownership – it is their system and, as their system produces better and better outputs, they derive a high degree of satisfaction (as well as looking good amongst their peers!)
- It enables a company to compete based on the quality of its experts. This is particularly significant in areas like diagnostic services, where although there are guidelines on test interpretations, a local expert can provide something much more specific to a clinician customer, giving a unique value-add.

Software

Pacific Knowledge System technology includes a complete suite of modules needed to deploy a sophisticated RippleDown system. These modules include:

- The RippleDown server (inferencing)
- The RippleDown Validator (monitoring)
- The RippleDown Knowledge Builder (knowledge acquisition)
- The RippleDown Translator (multi-lingual applications)
- The RippleDown Administrator (management of the Knowledge Bases and users)
- The RippleDown Interface Gateway (a wide range of communications plug-ins to interface with data repositories or other online information systems).

The deployment of RippleDown only requires standard computing platforms. It is highly portable as it is a pure Java implementation. The system can be deployed as a stand-alone application, a client-server architecture or in the cloud, pulling data from multiple sources.

Typical performance on a knowledge base with thousands of rules is about 300 transactions processed per second, making RippleDown an appropriate choice for applications where a sophisticated expert system is needed in a high transaction environment and where reliability is critical.