Rules Fest 2010

International Conference on Reasoning Technologies

October 11-14 • San Jose, CA USA

Grindwork Corporation
Intelligent Automation

JBoss by Red Hat

IBM

Production Systems Technologies

morris technical solutions

VisionArts communications
strategic thinking • creative action

Integrating Classic Rule Engines with Modern Scalable Architectures

Chandra Mouleeswaran

Founder, Symerica Systems LLC

chandra@symerica.com

Background

- AI professional with 25 years experience from: large multinational, Wall Street brokerage house and several startups
- Founded Symerica Systems LLC
  - Use of *Sym*bolic logic and num*erica*l data to build solutions
  - Technology company focused on developing products for real-time applications using rule engines, machine/statistical learning, web crawler, NLP etc
  - Use of standardized and scalable architectures
  - Knowledge engineering services

Outline of Presentation

- Review of past projects
- Rationales for applying rules
- Key observations and lessons learned
- Architectural preferences for real-time applications
- Current project
- Brief demo
- Conclusions

Past Projects

Domain	Technology	Customer Profile	Approximate Lifetime	Comments
Process Planning	Rules & Taxonomies	MNC	4 years	Difficulties with 3D shapes; Surprising ML experiments
Chip Partitioning Expert	OOP, Heuristic Search, Minimal Rules	MNC	> 4 years	Fewer experts; Capture of repetitive design patterns
Diagnosis of CT Scanners & Telecommunication Switches	Rules, Domain Specific Shell, TMS	MNC	4 years	KB maintenance using Machine Learning to process exceptions; Regression testing
Network Monitoring and Capacity Planning	**Rules, Real Time Data Acquisition, Statistical Trends**	**Financial/ Brokerage**	**> 14 years and currently in use**	**Custom-built rule engine; Technology transferred to non-AI staff**
Customs Clearance and Compliance; Workflow	Rules, J2EE	Startup	3 years	Rule engine for process understanding; Some parts replaced with procedural code

Power of Rule Engines

- Rule based approach solves a problem incrementally and increases our understanding of the domain. Declarative nature simplifies many tasks and keeps the code closer to our thought process
- Lends to rapid prototyping
- Extending coverage easily accomplished using modifiable facts in generic rule containers
- Being able to create intermediate facts at will is a very powerful mechanism in real time programming
- Well thought out rule systems have substantial lifetime

Key Elements for Successful Launch

- Real problem with no clear algorithmic solution
- Solid commitment from management
- High frequency of use
- Significant revenue impact
- Overwhelming amount of real time data (information overload) and long resolution times
- Traditional tools offer piece-meal solutions. Hard to integrate
- Fewer experts who get bogged down by every outage
- Small team with two engineers, specialized in the domain and AI, to built the prototype and demonstrate on a subset of the problem

Key Elements for Success Launch (contd.)

- Ease of maintenance and extension
- Ownership of software. Longest lifetime in custom-built engines that keep only the essential portions of control and integrate easily into working environment
- Simple conflict resolution – compilation ordering in rule files
- Efficiency - execution speed to match environment needs
- Good documentation
- Technology transfer to non-AI personnel
- Specialized Knowledge Vs Community Knowledge

Potential Impediments

- AI skeptics
- Low-level system experts with legacy code, resisting changes
- Environment where DB plays the dominant role, hard to comprehend the need for rule engines
- Plethora of tools, each addressing a small part of an overall problem, lacking an integrated view and giving a false sense of security
- Importance of knowledge engineering misunderstood. Rule engine projects treated as any other software project without identifying experts and involving in knowledge engineering sessions
- Unavailability of experts

Potential Impediments (contd.)

- Failure to integrate with working environment. One of the biggest hurdles in rule engine development is to bring all the data into a framework for rules to inspect. Many agents may be needed to solve this
- Shift in thinking, from the domain to programming language features and other interface details not directly related to the problem
- Cost of software, consultation and licensing
- Business models that don't do technology transfer and train non AI personnel

Project Initiation

- It may appear as though there are no rules of thumb available to solve a problem
- Knowledge acquisition is not always explicit. Sometimes, need to learn by doing or working with the experts
- Be very suspicious of a domain that claims to have millions of rules. It may actually be a small set of rules with millions of facts that instantiate in various situations. Look to generalizing rules (with variables) and minimizing one-shot rules (non chaining)
- Decide whether the default control strategy is acceptable. In many cases, applications might need a TMS rather than agenda-based conflict resolution

AI for Equity Research & Trading – Motivation

- Investors losing faith in Buy and Hold strategy
- User wants to make informed judgments
- Each investor has his/her own style and there is a need to capture the personalized “equity universe” and policies
- Current tools present overwhelming and conflicting data. Several accesses (Web Page or other) required to build up some understanding of the current state of an equity
- Following any particular research product finally leads to a dead-end from which there is no graceful way to continue the exploration. Stock screening and other tools are not state preserving
- Strategies are difficult to program and are not easy to blend with subjective rules
- Cognitive gap between data accumulation/review and trade execution

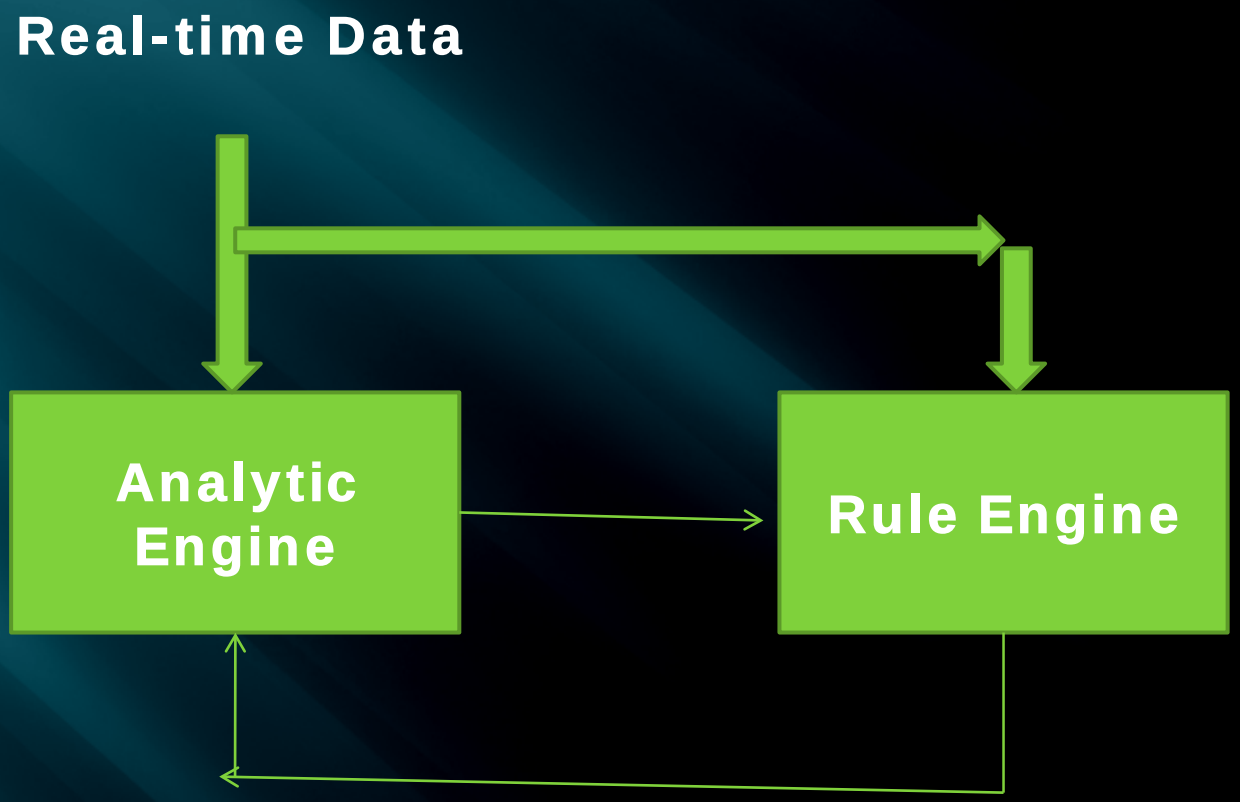
Trading & Equity Research (AI View)

- Assume user is reasonably savvy in trading stocks. The aim is to shift some of the cerebral load to the AI system
- No need to “seek out” information – plenty of online resources available. This has traditionally been a bottleneck for building expert systems. With financial domain, the data lands on your lap
- Rules create a hybrid approach using the best of both Tech and Fundamental Analyses
- Goal: Given a certain principal amount P and a set of possible candidates (equities), what would be right mix of candidates to realize near-term gain with lowered risk

Problem Features

- A certain amount of static knowledge and lot of data
- Both real-time and batch execution are required for meaningful solution
- Parameter acquisition and metrics collection based on large accumulation of data, and may depend on the equity
- Need for fine tuning parameters on a regular basis. Turn around time for fine tuning is short
- Exceptions can occur. Need to reschedule, realign information to stay on target
- Memory leak of a different kind occurs in rule engines – monotonic increase in assertions that go out of scope

Preferred Architecture



Rule Engine Details

- Rete-based rule engine
- Common base with customizable metrics and rules for individual users
- The inputs include market data, balance sheets, quarterly reports, past trading records, user notes from sessions etc
- User configurable data sources and taxonomies
- User configurable collection rates and distribution of data collectors
- User annotations following system recommendations

Rule Engine Details (contd.)

- Rules are not elegant way to represent a deterministic high-level sequence of actions (or workflow), without tampering with precedence/salience. Top-level control in engine based on an heuristic AI Algorithm
- Rules attached to various steps modify the state of an equity. Equities remain in various states and also post themselves in various decision layers {buy, potential buy, favorites, sell etc. }
- State evaluation is the weighted sum of many parameters, technical and fundamental included

Analytic Engine Details

- **Purpose of Analytic Engine is to generate statistical "facts" that may be used in the next cycle of reasoning**
- **Some of the analytical tools: simple statistical parameters, generalized price envelopes, linear regression, clustering etc**
- **Engine generates many levels of statistics, some overnight and some in batched processes during trading hours**

Aging

- Inputs include recommendations not acted upon by user
- Feedback analysis shows consistently poor performance predictions for some rules and thresholds

User Acceptance

- Delivering tools at the lowest common denominator: HTML/XML and Browsers
- Keeping interaction simple and presenting information that is cogent and lucid
- User is only interested in a subset of stocks after initial analysis

Brief Demo

- Ranked recommendations
- View into supporting rules
- Metrics
- Audit trails

Conclusions

- Rete-based rule engines are a natural fit for complex and ill structured real-time applications that have rules-of-thumb and large amount of rapidly growing data
- Combining rule engine with analytic engines will yield a powerful and adaptive framework to address these problems
- Scalable architectures support batch-processing of large datasets
- In general, rule engines are under-utilized due to lack of awareness and resistance from legacy systems. Solid management commitment is needed to overcome these barriers
- Knowledge engineering, integrating with existing infrastructures, handling exceptions, improving performance over time and user acceptance will be the key challenges