Learning inside the Rete algorithm

YKY (Yan King Yin)
general.intelligence@gmail.com

February 17, 2019

Abstract

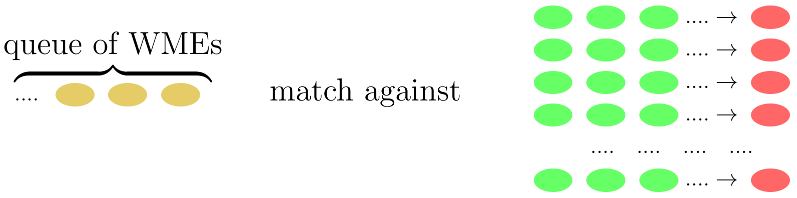
1 Introduction to the Rete algorithm

The general form of a **production rule** is like this:

$$\overbrace{\text{🟢} \wedge \text{🟢} \wedge \text{🟢} ....}^{\text{condition}} \rightarrow \overbrace{\text{🔴}}^{\text{action}} \tag{1}$$

where ∧ denotes logical conjunction (AND).

Typically, we would be trying to match a relatively small number of **facts** (that represent the current **state**, or **working memory**) against a very large number of **rules**:



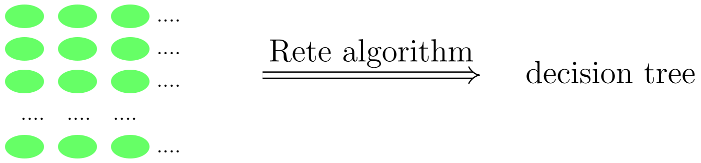
(2)

where 🟡 = WME = working memory element = fact = **grounded** logic formula = formula not containing variables.

Obviously, if the number of rules is large, it would be time-consuming to test each rule one by one to see if they apply.

It would be much more efficient if we could look at each 🟡 and immediately see which rule(s) may apply to it. This is the idea behind Rete.

In other words, we would like to **compile** the rule conditions 🟢 into a **decision tree**:



(3)

The actions 🔴 of the rules do not figure in the decision process.

1.1 How the Rete graph is constructed

α-nodes:

$$\overbrace{\color{green}\bullet}^{\alpha_1} \wedge \overbrace{\color{green}\bullet}^{\alpha_2} \wedge \overbrace{\color{green}\bullet}^{\alpha_3} \,.... \to {\color{red}\bullet} \tag{4}$$

β-nodes:

$$\overbrace{\overbrace{\overbrace{\color{green}\bullet}^{\beta_1} \wedge {\color{green}\bullet}}^{\beta_2} \wedge {\color{green}\bullet}}^{\beta_3} \,.... \to {\color{red}\bullet} \tag{5}$$

2 Relation to deep learning

A typical **neural network** is:

$$F(\vec{x}) = \oint(\overset{\textbf{weights}\text{ matrix}}{W_1}\oint(W_2 \ldots \oint(W_{\overset{\text{total \# layers}}{L}}\ \vec{x}))) \tag{6}$$

Its set of **parameters** is $\Theta = \{W^{\ell}_{i,j}\} \in \mathbb{R}^m$, where $m = \#$ weights.