Santa's Reindeer - The Rules

Corticon Rule Modeling Challenge Xmas 2014 Santa's Reindeer

Santa always leaves plans for his elves to determine the order in which the reindeer will pull his sleigh. This year, for the European leg of his journey, his elves are working to the following schedule, which will form a single line of nine reindeer

The Rules

1. Comet behind Rudolph, Prancer and Cupid.
2. Blitzen behind Cupid
3. Blitzen in front of Donder, Vixen and Dancer.
4. Cupid in front of Comet, Blitzen and Vixen.
5. Donder behind Vixen, Dasher and Prancer.
6. Rudolph behind Prancer
7. Rudolph in front of Donder, Dancer and Dasher.
8. Vixen in front of Dancer and Comet.
9. Dancer behind Donder, Rudolph and Blitzen.
10. Prancer in front of Cupid, Donder and Blitzen.
11. Dasher behind Prancer
12. Dasher in front of Vixen, Dancer and Blitzen.
13. Donder behind Comet and Cupid.
14. Cupid in front of Rudolph and Dancer.
15. Vixen behind Rudolph, Prancer and Dasher.

Create a rule model that determines the order of the reindeer.

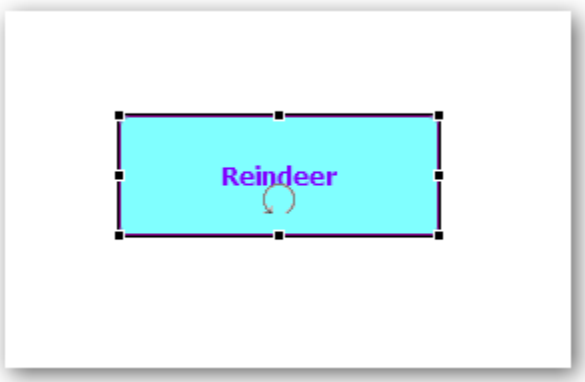
The Approach

The 12 statements above are effectively the rules that solve the problem.

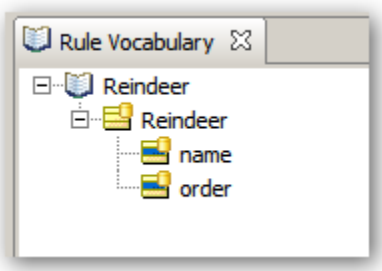
The challenge is to find a way to represent them in Corticon such that we can use the power of Corticon's inferencing to continue working until all the reindeer are in the correct order.

The basic idea is to assign a sequence number to each reindeer (initially one).

Then compare reindeer (two at a time) and use the rules to adjust that sequence number whenever reindeer are found to out of order. Because modifying a reindeer's sequence number may cause that reindeer to break other rules we'll need to repeat the process of applying the rules until no more changes are made to the sequence numbers. Then we will have a solution. This means we will need to make the rule sheet iterate. This is done by drawing the rule flow diagram and marking the rule sheet with the iterate symbol:



Vocabulary



Rule Sheet

Conditions	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reindeer A	{'Prancer', 'Cupid', 'Rudolph'}	'Cupid'	'Blitzen'	'Cupid'	{'Prancer', 'Dasher', 'Vixen'}	'Prancer'	'Rudolph'	'Vixen'	{'Rudolph', 'Blitzen', 'Donder'}	'Prancer'	'Prancer'	'Dasher'	{'Cupid', 'Comet'}	'Cupid'	{'Prancer', 'Rudolph', 'Dasher'}
Reindeer B	'Comet'	'Blitzen'	{'Vixen', 'Donder', 'Dancer'}	{'Blitzen', 'Vixen', 'Comet'}	'Donder'	'Rudolph'	{'Dasher', 'Donder', 'Dancer'}	{'Comet', 'Dancer'}	'Dancer'	{'Cupid', 'Blitzen', 'Donder'}	'Dasher'	{'Blitzen', 'Vixen', 'Dancer'}	'Donder'	{'Rudolph', 'Dancer'}	'Vixen'
Is A behind (or at the same place) as B. (i.e. is the sequence number of A >= the sequence number of B)	T	T	T	T	T	T	T	T	T	T	T	T	T	T	T
Actions															
Post Message(s)															
Move reindeer B one place down the line (increase the sequence number)	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑

Scope

- Reindeer [R1]
- Reindeer [R2]

	Conditions	1	2	3	4	5	6	7
a	R1.name	{'Prancer', 'Cupid', 'Rudolph'}	'Cupid'	'Blitzen'	'Cupid'	{'Prancer', 'Dasher', 'Vixen'}	'Prancer'	'Rudolph'
b	R2.name	'Comet'	'Blitzen'	{'Vixen', 'Donder', 'Dancer'}	{'Blitzen', 'Vixen', 'Comet'}	'Donder'	'Rudolph'	{'Dasher', 'Donder', 'Dancer'}
c	R1.order >= R2.order	T	T	T	T	T	T	T
	Actions							
	Post Message(s)							
A	R2.order+=1	☑	☑	☑	☑	☑	☑	☑

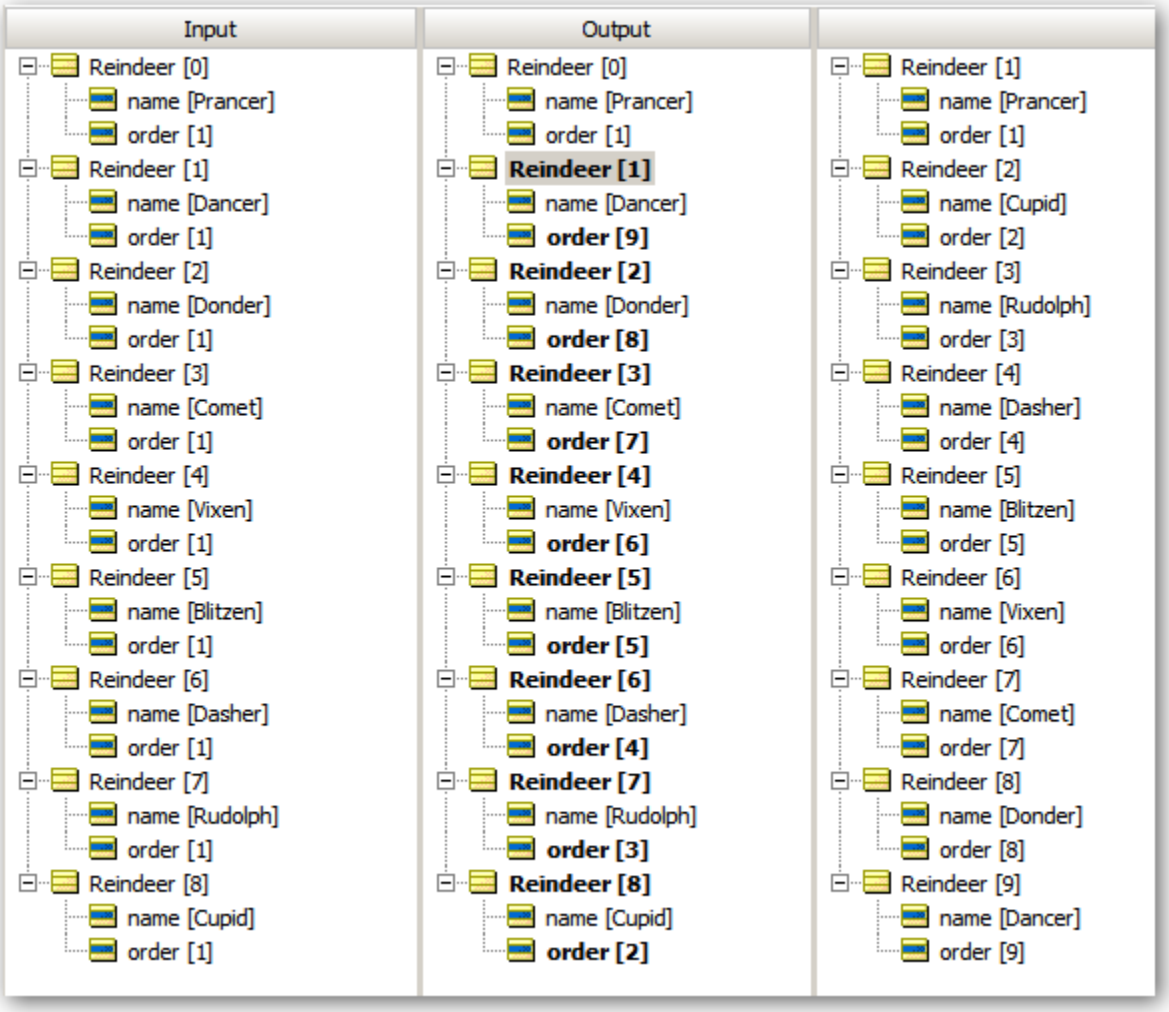
How it Works

By specifying two aliases to the entity Reindeer Corticon will automatically try every possible combination of two reindeer from the set of reindeer that is passed in as the data payload.

The rules then modify the sequence number of one of the reindeer if it is found to break the rules.

This process repeats until no more rules fire (i.e. no reindeer are out of sequence)

Test Case



Note that it's the value of the attribute called order that determines the sequence. The reindeer are not displayed in the tester in order. But the tester automatically compares the attributes of the reindeer and if there were any differences they would be highlighted in red.

Playing with the Rule Model

If we apply the Corticon compression feature to the rule sheet we'll see that Corticon is able to express the same logic in fewer rules. 10 rules instead of 15.

Rules 2, 4 and 14 can be merged since they all apply to cupid

Rules 6, 10 and 11 can be merged since the all apply to prancer.

Rules 5 and 13 can be merged since they all apply to donder.

Conditions	1	2	3	4	5	6	7	8	9	10
R1.name	{'Cupid', 'Prancer', 'Rudolph'}	'Cupid'	'Blitzen'	{'Comet', 'Cupid', 'Dasher', 'Prancer', 'Vixen'}	'Prancer'	'Rudolph'	'Vixen'	{'Blitzen', 'Donder', 'Rudolph'}	'Dasher'	{'Dasher', 'Prancer', 'Rudolph'}
R2.name	'Comet'	{'Blitzen', 'Comet', 'Dancer', 'Rudolph', 'Vixen'}	{'Dancer', 'Donder', 'Vixen'}	'Donder'	{'Blitzen', 'Cupid', 'Dasher', 'Donder', 'Rudolph'}	{'Dancer', 'Dasher', 'Donder'}	{'Comet', 'Dancer'}	'Dancer'	{'Blitzen', 'Dancer', 'Vixen'}	'Vixen'
R1.order >= R2.order	T	T	T	T	T	T	T	T	T	T
Actions										
Post Message(s)										
R2.order+=1	☑	☑	☑	☑	☑	☑	☑	☑	☑	☑