In this document, I give some examples of first-order definitions and how do they compose. I took the previous examples of Tobias, and ignored for now semiring values. Thus, soft constraints are considered as constraints.

From automata to rules

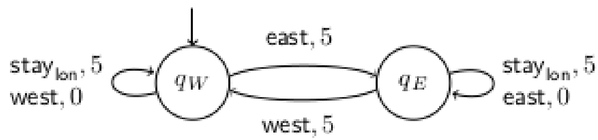

Figure 1: A component modeling a patrolling movement.

According to the Component Action System, all actions of the soft constraint can be represented by a predicate over a set of ports. Define a variable s that indicates the current state. In A_1 automaton in Figure 1, $s_1 = 1$ refers to state q_W and $s_1 = 2$ to state q_E.

Now, the automaton can be viewed as a single first-order formula :

$$\begin{aligned}\phi(A_1) = \quad & (s_1 = 1 \wedge ((east \wedge s'_1 = 2) \vee (west \ \wedge s'_1 = 1) \vee (stay_{lon} \wedge s'_1 = 1))\vee \\ & (s_1 = 2 \wedge ((east \wedge s'_1 = 2) \vee (west \wedge s'_1 = 1) \vee (stay_{lon} \wedge s'_1 = 2))) \\ = \quad & r_1 \vee r_2 \vee r_3 \vee r_4 \vee r_5 \vee r_6,\end{aligned}$$

with
$r_1 := s_1 = 1 \wedge (east \wedge s'_1 = 2)$
$r_2 := s_1 = 1 \wedge (west \ \wedge s'_1 = 1)$
$r_3 := s_1 = 1 \wedge (stay_{lon} \wedge s'_1 = 1)$
$r_4 := s_1 = 2 \wedge (east \wedge s'_1 = 2)$
$r_5 := s_1 = 2 \wedge (west \wedge s'_1 = 1)$
$r_6 := s_1 = 2 \wedge (stay_{lon} \wedge s'_1 = 2)$

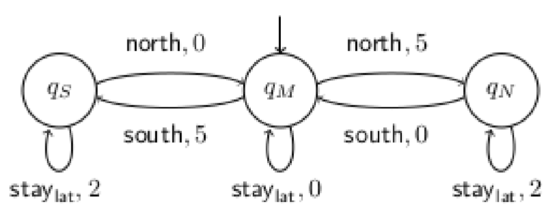

Figure 2: A component modeling divergence from the path in two directions.

Similar to A_1 in Figure 1, automaton A_2 in Figure 2 can also be turned into a first-order formula. In automaton A_2, $s_2 = 1$ refers to state q_M, $s_2 = 2$ refers to state q_S and $s_2 = 3$ refers to the state q_N.
The corresponding formula is

$$\begin{aligned}\phi(A_2) = \quad & (s_2 = 1 \wedge ((south \wedge s'_2 = 2) \vee (stay_{lat} \wedge s'_2 = 1) \vee (north \wedge s'_2 = 3)) \\ & (s_2 = 2 \wedge ((north \wedge s'_2 = 1) \vee (stay_{lat} \wedge s'_2 = 2)) \\ & (s_2 = 3 \wedge ((south \wedge s'_2 = 1) \vee (stay_{lat} \wedge s'_2 = 3)) \\ = \quad & p_1 \vee p_2 \vee p_3 \vee p_4 \vee p_5 \vee p_6 \vee p_7,\end{aligned}$$

with
$p_1 := s_2 = 1 \wedge (south \wedge s'_2 = 2)$

$p_2 := s_2 = 1 \wedge (north \wedge s_2' = 3)$
$p_3 := s_2 = 1 \wedge (stay_{lat} \wedge s_2' = 1)$
$p_4 := s_2 = 2 \wedge (north \wedge s_2' = 1)$
$p_5 := s_2 = 2 \wedge (stay_{lat} \wedge s_2' = 2)$
$p_6 := s_2 = 3 \wedge (south \wedge s_2' = 1)$
$p_7 := s_2 = 3 \wedge (stay_{lat} \wedge s_2' = 3)$

Composition of automata

The composition of A_1 and A_2 gives a new Component Automaton with 6 states (cartesian product of states from A_1 and A_2) and 42 transitions. From the logical perspective, composition only applies a composition operator on the formula of A_1 and the formula of A_2. For join composition, the operator is normal conjunction. The intuition behind this conjunction is that the new automaton must, for each transition, take a transition of A_1 and A_2. Then, assuming that automata are in disjunctive normal form and each clause represents a transition, a transition of A is a transition of A_1 (one of its clauses) and a transition of A_2 (i.e. the conjunction of the two clauses).

For join composition, with $R = \{r_1, ..., r_6\}$ and $P = \{p_1, ..., p_7\}$:

$$\begin{aligned}\phi(A) = &\quad \phi(A_1) \wedge \phi(A_2)\\ \equiv &\quad (r_1 \vee r_2 \vee r_3 \vee r_4 \vee r_5 \vee r_6) \wedge (p_1 \vee p_2 \vee p_3 \vee p_4 \vee p_5 \vee p_6 \vee p_7)\\ = &\quad \bigvee_{r_i \in R} r_i \wedge \bigvee_{p_i \in P} p_i\\ = &\quad \bigvee_{r_i \in R, p_i \in P} r_i \wedge p_i\end{aligned}$$

More generally, for Soft Constraint Automata, the composition operator can be seen as a polymorphic operator $\otimes$, that behaves like a conjunction, except for soft constraints.

$$\begin{aligned}\phi(A) = &\quad \phi(A_1) \otimes \phi(A_2)\\ = &\quad \bigvee_{r_i \in R} r_i \otimes \bigvee_{p_i \in P} p_i\\ = &\quad \bigvee_{r_i \in R, p_i \in P} r_i \otimes p_i\end{aligned}$$

We assume that $\otimes$ distributes over $\wedge$, hence the first product $r_1 \otimes p_1$ is :

$$\begin{aligned}r_1 \otimes p_1 = &\quad (s_1 = 1 \wedge (east \wedge s_1' = 2)) \otimes (s_2 = 1 \wedge (south \wedge s_2' = 2))\\ = &\quad s_1 = 1 \otimes s_2 = 1 \wedge (east \wedge s_1' = 2) \otimes (south \wedge s_2' = 2)\\ = &\quad s_1 = 1 \wedge s_1' = 2 \wedge s_2 = 1 \wedge s_2' = 2 \wedge east \otimes south\end{aligned}$$

The polymorphic operator $\otimes$ should evalute to a different operator regarding the type of the actions. In this example, if the type of action *east* is Longitude and the type of action *south* is Latitude, by defining a type hierarchy, $\otimes$ has different interpretations. If Longitude is prefered as Latitude, $\otimes$ would be evaluated to $\rhd$. If the types are equals, then $\otimes$ is evaluated to $\wedge$.

Composition must also take care of semiring value composition. At this point, I don't see yet how to properly integrate soft constraint inside those formula.