Matching Activity Conditions to States

Matching conditions of an activity to a state depends on how containment and connectivity are defined in both.

Matching is done as follows:

1. Conditions in an activity are converted into Bigraph objects.
   1. The definition of a condition follows the one described by the incident meta-model.
   2. XML representation of a condition is converted to JSON object, which is then converted to Bigraph object.
2. The Bigraph object of a condition is compared against the signature of the Bigraph.
   1. If a condition has more connections for an entity than that in the signature (i.e. the number of defined *outernames* in a condition is more than that in the signature), then ***matching fails for all states***.
      1. For example, if in a condition an entity (or child) defines *three outernames* (i.e. connections) and in the bigraph signature the corresponding control to this entity defines only two outernames, then matching fails for all states.
   2. If a condition has less connections for an entity, then:
      1. If knowledge about the connections is complete, then ***matching fails for all states***.
      2. If knowledge is partial, then the rest are defined as links (i.e. XX:**e**) in the bigraph object of the condition.
3. Each condition is then compared against each state (already converted to Bigraph objects) using the bigraph matching, which is implemented by the method *match* that is provided by the *LibBig* library.
4. If the *match* method returns any object in the iterator (it returns an iterator, which shows where in a state the condition is match. Currently we are interested if there are anything return, we are not interested how many and where in a state), then ***there is a match*** and the state number is added to the condition (i.e. *predicate* object). Otherwise (i.e. if the iterator is empty) then ***there is no match*** between the condition and the current state.

Using the [match] function in the LibBig library:

The function depends in matching mainly on the containment relationships between the different entities. So it is important to specify *sites* where knowledge is not exact. For example, if a *Room* is an entity but we have no exact knowledge on what is inside the *Room* except a *SmartLight*, then it is necessary to add a *site* when creating a condition using the incident pattern meta-model.

For connectivity, matching is partial. For example:

* if an entity has 4 outernames (as arity in the signature) and in a condition there are 2 outernames defined and knowledge is partial and the rest are not *closed*, then defining only the 2 outernames by the Bigraph object will make the other two outernames *links* (i.e. XX:**e**), hence, it ***won’t match*** with the states that specify all 4 outernames.
  + Outernames by signature: 4
  + Outernames defined in condition: 2
  + Knowledge: partial, then:
    - we can define the other two outer name when creating the Bigraph object. In this situation, the condition ***will match*** with all states that define the 4 outernames as such (non are closed). It won’t matter that in a state outernames of an entity is associated with others. **What if in a state some outernames are closed (not found in the bigraph object)?**
    - If the remaining two outernames are not defined then they are defined automatically by the Bigraph class as links (i.e. XX:**e**). in this case, they ***will not match*** with states that define outernames as XX:**o** (i.e. as outernames).
    - If we define the remaining outernames as closed, then they will not match to states that define their outernames. **Will they match with states that close two of their outernames?** Probably?