TSC Binding Engine

latest: 20050211

Background

TSC fires process rules against episodes. An episode is a kind of stage, snapshot, or environment in which all that is known to be important about some process is contained.

Rules

A rule takes the form:

    ifActors (list of actors)
    ifRelations (list of relations)
    ifStates (list of states)
    thenActors (list of new actors)
    thenRelations (list of changed or new relations)
    thenStates (list of changed or new stataes)

Where the lists take the form of a sentence:

    (predicate (subject, optionalObject) truth)

In rules, the subject and object (if used), take the form of a variable

    e.g. *foo

Episodes

An episode takes the form:

    actors (list of actors)
    relations (list of relations)
    states (list of states)

where the lists take the same sentence form, with the exception that subject and object represent objects which exist in the environment.

Binding

Binding simply means matching unbound variables to existing objects, creating bound variables.

Here are some traces from TinyTSC:

TR 2: [[stem.cell(stem.cell1 |)], [helper.t-cell(helper.t-cell1 |)], [macrophage(Macrophage1 |)], [b-cell(B-cell1 |)], [activated.b-cell(activated.b-cell1 |)], [act.h.t-cell(act.h.t-cell1 |)], [gm-csf(csf1 |)], [il(il1 |)], [il-2(il2 |)], [virus(virus1 |)], [antigen(antigen1 |)], [cytokine.receptor(cytokine.receptor1 |)], [plasma.cell(plasma.cell1 |)], [memory.cell(memory.cell1 |)], [blast.cell(blast.cell1 |)], [antibody(antibody1 |)]]

TR 3: [[antigen(*antigen |)], [b-cell(*b-cell |)], [helper.t-cell(*helper.t-cell |)], [macrophage(*macrophage |)]]

TR 2 represents some sentences from the actors field of an episode.

TR3 represents some unbound variables from a particular rule.

On inspection: notice these candiate bindings:

[antigen(antigen1 |)] – [antigen(*antigen |)]

[b-cell(B-cell1 |)] -- [b-cell(*b-cell |)]

[helper.t-cell(helper.t-cell1 |)] – [helper.t-cell(*helper.t-cell |)]

[macrophage(Macrophage1 |)] – [macrophage(*macrophage |)]

Consider this scenario:

An episode has 3 actors:

(antigen (ant1) true) (antigen (ant2) true) (b-cell (cellX) true)

A rule has 3 ifActors:

(antigen (*a1) true) (antigen (*a2) true) (b-cell (*c1) true)

The trick is to create just one binding for *a1 and one for *a2.

We don't really care which order they come in; we only care that each gets a unique binding:

*a1 – ant1 *a2 – ant2 *c1 – cellX