pyactr_on_google_colab

May 19, 2019

```
In [2]: !pip3 install pyactr
```

```
Collecting pyactr
  Downloading https://files.pythonhosted.org/packages/40/ff/56194da27074e31fe098f27624dfac73dc
     || 61kB 18.5MB/s
Requirement already satisfied: pyparsing in /usr/local/lib/python3.6/dist-packages (from pyact
Requirement already satisfied: numpy in /usr/local/lib/python3.6/dist-packages (from pyactr) (
Collecting simpy (from pyactr)
  Downloading https://files.pythonhosted.org/packages/5a/64/8f0fc71400d41b6c2c6443d333a1cade45
Installing collected packages: simpy, pyactr
Successfully installed pyactr-0.2.4 simpy-3.0.11
```

```
In [0]: import pyactr as actr
```

```
In [5]: actr.chunktype("word", "meaning, category, number, synfunction")
        actr.chunktype("goal_lexeme", "task, category, number")

        carLexeme = actr.makechunk(
            nameofchunk="car",
            typename="word",
            meaning="[[car]]",
            category="noun",
            number="sg",
            synfunction="subject")

        agreement = actr.ACTRModel()

        dm = agreement.decmem
        dm.add(carLexeme)

        agreement.goal.add(actr.chunkstring(string="""
            isa goal_lexeme
            task agree
            category verb"""))

        agreement.productionstring(name="retrieve", string="""
            =g>
            isa goal_lexeme
            category verb
            task agree
            ?retrieval>
            buffer empty
            ==>
            =g>
            isa goal_lexeme
            task trigger_agreement
            category verb
            +retrieval>
            isa word
            category noun
            synfunction subject
            """)

        agreement.productionstring(name="agree", string="""
            =g>
            isa goal_lexeme
            task trigger_agreement
            category verb
            =retrieval>
            isa word
            category noun
            synfunction subject
            number =x
            ==>
            =g>
            isa goal_lexeme
            category verb
            number =x
            task done
            """)

        agreement.productionstring(name="done", string="""
            =g>
            isa goal_lexeme
            task done
            ==>
            ~g>""")
```

```
Out[5]: {'=g': goal_lexeme(category= , number= , task= done)}
        ==>
        {'~g': None}
```

```
In [6]: agreement_sim = agreement.simulation()
        agreement_sim.run()
        print("\nDeclarative memory at the end of the simulation:")
        print(dm)
```

```
(0, 'PROCEDURAL', 'CONFLICT RESOLUTION')
(0, 'PROCEDURAL', 'RULE SELECTED: retrieve')
(0.05, 'PROCEDURAL', 'RULE FIRED: retrieve')
(0.05, 'g', 'MODIFIED')
(0.05, 'retrieval', 'START RETRIEVAL')
(0.05, 'PROCEDURAL', 'CONFLICT RESOLUTION')
(0.05, 'PROCEDURAL', 'NO RULE FOUND')
(0.1, 'retrieval', 'CLEARED')
(0.1, 'retrieval', 'RETRIEVED: word(category= noun, meaning= [[car]], number= sg, synfunction=
(0.1, 'PROCEDURAL', 'CONFLICT RESOLUTION')
(0.1, 'PROCEDURAL', 'RULE SELECTED: agree')
(0.15, 'PROCEDURAL', 'RULE FIRED: agree')
(0.15, 'g', 'MODIFIED')
(0.15, 'PROCEDURAL', 'CONFLICT RESOLUTION')
(0.15, 'PROCEDURAL', 'RULE SELECTED: done')
(0.2, 'PROCEDURAL', 'RULE FIRED: done')
(0.2, 'g', 'CLEARED')
(0.2, 'PROCEDURAL', 'CONFLICT RESOLUTION')
(0.2, 'PROCEDURAL', 'NO RULE FOUND')

Declarative memory at the end of the simulation:
{word(category= noun, meaning= [[car]], number= sg, synfunction= subject): array([0.]), goal_l
```

```
In [0]:
```