The ACT-R 6.0 temporal module

Niels Taatgen, Hedderik van Rijn & John R. Anderson

17 November 2005

1 Introduction

At the ICCM 2004 conference, Taatgen, Van Rijn and Anderson introduced an additional ACT-R 5.0 module for time perception (the Temporal buffer). This document describes the update of that module for ACT-R 6. The idea behind this module is that time can be represented by "ticks" which can be thought of as markers that indicate how much time has passed, and which are distributed in time with increasingly wider intervals. This results in high resolution time estimation for short intervals, but less accurate timing for longer intervals. The conceptual underpinnings and additional empirical data are presented in the ICCM'04 paper (`http://www.ai.rug.nl/~niels/temporal.html`) and a CogSci '05 paper (`http://www.ai.rug.nl/~niels/publications/dual-task-time-n.pdf`). This document describes how to use the ACT-R implementation for the Temporal module that can be found at `http://www.ai.rug.nl/~niels/temporal.html`.

This document describes the ACT-R 6 version of the module, there is a separate ACT-R 5 version available on the same webpage.

2 Installation

Copy the file `temporal.lisp` into the `modules` directory of ACT-R 6.

3 Structure of the Temporal Buffer

The Temporal buffer contains a single slot: `ticks`. The value of this slot is an integer representing how many ticks have passed since last reset of the Temporal buffer. This value is automatically updated as time passes.

4 Operations on the Temporal Buffer

4.1 Starting the timer

By issuing a `+temporal> isa time` on the RHS of a production, the Temporal buffer, that is, the current tick count, is initialized to zero. A chunk of type `time` is placed into temporal buffer with the tick slot set to 0. The Temporal module will now start incrementing the ticks at certain moments in time as described below.

4.2 `=temporal>`

Using `=temporal>` on the LHS of a production can be used to harvest the current ticks:

```
(p harvest-temporal
  =goal>
    isa     demo-time
    curtime nil
  =temporal>
    isa     time
    ticks   =ticks
 ==>
  =goal>
    curtime =ticks)
```

This production copies the current value in the temporal buffer into the goal. Another use of the buffer in the condition of a rule is to match the current tick value against a set value:

```
(p match-temporal
  =goal>
    isa     demo-time
    curtime =ticks
  =temporal>
    isa     time
    ticks   =ticks
 ==>
  !output!  ("~S ticks have passed!~%" =ticks))
```

This production will fire at the moment that the time reaches the number stored in the goal.

The Temporal buffer does not use strict harvesting, so matching the buffer will not clear it.

4.3 Stopping the timer

The `+temporal> isa clear` operation clears the Temporal buffer, removing the current time representation from the Temporal buffer. Moreover, it also turns off the ACT-R internal calculation of time ticks. It can therefore be economical to issue a clear command when a model does (temporarily) not use the Temporal module.

5 Temporal Module Parameters

The length of the ticks is defined as $t_n = a \cdot t_{n-1} +$ act-r-noise(mean=0, $sd = b * a \cdot t_{n-1}$), given that the first tick is $t_0 =$ starttick + act-r-noise(mean=0, $sd = 5b \cdot$ starttick). Note that is is different from the ICCM paper, in which the starttick was refered to as being constant instead of noisy. The value of t_0 is recomputed after a reset of ACT-R, but is left the same on a new Temporal module request.

The table below links the equation's parameters to ACT-R variables that can be set with the `sgp` command. We strongly recommend keeping the parameters at their default values.

Equation	Default value	Variable
a	1.1	`:time-mult`
b	0.015	`:time-noise`
starttick	1.1	`:master-start-increment`

HvR, 050315, updated NT, 051117