Tracker Module

The tracker module provides the model with a system that is able to estimate the best value for a quantity based on one or two feedback signals, a “good” result and a “bad” result, which may be noisy, have different weights, and which may not be directly associable to the actions which caused them. That estimated quantity is updated periodically by the tracker module for the model to use without any additional actions needed by the model, and multiple different quantities may be estimated simultaneously by the tracker module (each quantity will be estimated using a separate ‘tracker’).

This is currently implemented as a pseudo-“rational” function estimator which tries to fit a 3-parameter quadratic function relating the control value to the feedback. The choice of a quadratic function reflects a non-informed guess as to how learning extrapolates over the range of possible values. This is the process which it uses to do so:

1. When a tracker is created it starts with an initial evaluation that is uniform over the range of control values and chooses one at random from that range.

2. It determines the length of the update interval from an exponential distribution based on a mean duration which can be provided for the tracker (it defaults to 10 seconds). The choice of an exponential distribution for evaluation reflects a non-informed guess as to what the evaluation periods are. That interval is bounded to be at least 1 second, and no more than 60 seconds, and generated like this:

-d * log(random[0,1])

d is the mean duration

3. Good and bad events are recorded until the update interval completes.

4. It computes a mean rate of return from the good and bad events during the interval. That mean rate of return is added to the estimation of the quadratic function with a weight equal to the duration of the interval.

5. Using the updated estimate of the quadratic function it selects a new control value using a softmax rule. The temperature used in the softmax rule decreases as time progresses according to the function:

init-temp/(1 + Total-Time/Scale-time)

. Unless set otherwise the initial temperature (init-temp) is 1 and the time scale (scale-time) is 180 secs. This means that in 3-minute Space Fortress the temperature is 1 at the beginning of the first game, .5 at the beginning of the second game, .2 at the beginning of the fifth game, and .05 at the beginning of the 20 game.

6. Return to step 2.

The control value is made available to the model through the slot of a chunk in a buffer, and the slot and buffer are specified when a tracker is created. Each tracker must specify a unique slot for its control value. The good and bad feedback are also determined from slots of chunks in buffers which are specified when the tracker is created (both slots must be for a chunk in the same buffer). Whenever the chunk in a buffer which is associated with a tracker's feedback is set or modified through a scheduled event, the feedback will be recorded. If a chunk is set in the buffer then the value in both the good and bad slot (if available) are recorded. If a chunk is modified in the buffer only those slots which are modified are checked for a feedback event occurring.

After a tracker has been created, some of its components can be updated to adjust the process. When a modification happens that will initiate an immediate update to create a new control value.

Parameters

All of the default components of how a tracker operates can be specified with parameters. The tracker module can also provide a detailed trace of the actions it performs if the parameter to do so is enabled, and there are parameters which can be set to apply a decay function to the use when updating the equation.

:bad-scale

The bad-scale parameter can be set to a function, command name string, or **nil**. if it is set to a function or command string that function/command will be passed the value from the bad slot when it is changed and the value returned should be the number which indicates the value to apply when updating the estimate. The default value is **nil** which means there is no scaling applied.

:bad-weight

The bad-weight parameter can be set to a number, and that number is multiplied by the value of a bad feedback result (either the value from the slot or the result of the bad-scale if one exists) to produce the value for the event. The default value is -1.

:control-buffer

The control-buffer parameter can be set to the name of a buffer, and indicated the default buffer in which the slot for the control value will be found. The default value is goal.

:control-count

The control-count parameter indicates the default number of possible control results a tracker will use between the min and max values indicated (inclusive of both). The default value is 21.

:control-max

The control-max parameter indicates the default maximum value for the range of control results. The default value is 1.

:control-min

The control-min parameter indicates the default minimum value for the range of control results. The default value is 0.

:control-scale

The control-scale parameter can be set to a function, command name string, or **nil**. if it is set to a function or command string that function/command will be passed the estimated value for the control result when it is updated and the value returned will be the one placed into the control slot. The default value is **nil** which means there is no adjustment to the value applied.

:good-scale

The good-scale parameter can be set to a function, command name string, or nil. if it is set to a function or command string that function/command will be passed the value from the good slot when it is changed and the value returned should be the number which indicates the value to apply when updating the estimate. The default value is **nil** which means there is no scaling applied.

:good-weight

The good-weight parameter can be set to a number, and that number is multiplied by the value of a good feedback result (either the value from the slot or the result of the good-scale if one exists) to produce the value for the event. The default value is -1.

:initial-temp

The initial-temp parameter sets the value for the default initial temperature used by a tracker. The default value is 1.

:temp-scale

The temp-scale parameter sets the default value used in the computation of the temperature decrementing for a tracker. The default value is 180.

:trace-tracker

The trace-tracker parameter controls whether the module prints information in the model trace with all of the details about the tracker actions which occur. It can be set to **t** or **nil**, and the default value is **nil**.

:tracker-decay

The tracker-decay parameter sets the decay value used when discounting the previous results when updating the a tracker. It can be set to a number or **nil**. The default value is **nil**, but it will be set automatically to .5 or .99 if it is **nil** when decay is enabled with a power law or exponential decay mechanism respectively using the :tracker-decay-method parameter.

:tracker-decay-method

The tracker-decay-method parameter controls whether or not previous results are discounted when updating a tracker estimate. It can be set to **nil**, **power**, or **exponential**. If it is non-**nil** then the indicated decay mechanism is applied to the past estimates when updating a tracker. The default value is **nil**.

:update-delay

The update-delay parameter sets the default value for the mean of the update interval (in seconds) used by a tracker. The default value is 10.

:values-buffer

The values-buffer parameter can be set to the name of a buffer and indicates the default buffer in which the good and bad slots for a tracker will be located. The default value is imaginal.

Chunk-types & Chunks

The tracker module defines the following chunk-types:

```
(chunk-type tracker-params control-scale good-slot bad-slot good-weight bad-weight
            good-scale bad-scale min max control-count temp scale delay)

(chunk-type (tracker-modification (:include tracker-params)))
```

```
(chunk-type (track-outcome (:include tracker-params))
            control-slot good-buffer bad-buffer)
```

It creates no initial chunks.

Tracker buffer

The tracker buffer is a searchable multi-buffer. It holds chunks which represent the information for all of the trackers which have been created. Because each tracker must have a unique value for the control-slot that slot can be used as a way to find a specific tracker in a production.

Activation spread parameter: :tracker-activation
Default value: 0.0

Queries

The tracker buffer only responds to the default queries, and does not maintain any queryable state. The results for the standard queries will be:

'State busy' will always be **nil**.

'State free' will always be **t**.

'State error' will always be **nil**.

Requests

track-outcome

control-slot *control*
[**good-slot** *good-slot* | **bad-slot** *bad-slot* | **good-slot** *good-slot* **bad-slot** *bad-slot*]
{**good-buffer** *good-buffer*}
{**good-weight** *good-weight*}
{**good-scale** *good-scale*}
{**bad-buffer** *bad-buffer*}
{**bad-weight** *bad-weight*}
{**bad-scale** *bad-scale*}
{**min** *minimum-control-value*}
{**max** *maximum-control-value*}
{**control-count** *number-of-control-values*}
{**control-scale** *control-scale*}

{**temp** *initial-temperature*}
{**scale** *temperature-scale*}
{**delay** *update-delay*}

All requests to the tracker buffer create a new tracker. Each request must have a control slot which is unique among the trackers that have been created, and must have at least one of the good-slot or bad-slot values specified but may provide both. The rest of the slots control the details for the tracker, and any which are not provided will be set to the current value from the corresponding parameter of the module.

If all the provided parameters are valid a new tracker is created and a chunk which contains all of its information will be added to the set of chunks for the tracker buffer. It will then schedule an event to modify the control-slot of the chunk in the control-buffer with the initial guess for the control value:

```
0.050   TRACKER                 MOD-BUFFER-CHUNK GOAL
```

That event is scheduled at the time the tracker is created with a priority of -2000 so that other actions at that time e.g. other production requests, will have an opportunity to create the chunk in the control buffer before the tracker sets the initial value. If there is no chunk in the buffer when that modification happens then a warning will be printed and the initial value will not be set.

```
#|Warning: schedule-mod-buffer-chunk called with no chunk in buffer GOAL |#
```

When a tracker updates it will generate two or three events. There will always be an update-tracker event at the time of the update indicating the control slot that is being updated (only shown in high detail trace):

```
4.038   TRACKER                 update-tracker VALUE
```

If there is a chunk in the control buffer it will generate a mod-buffer-chunk event to update the control-slot's value with the new guess:

```
4.038   TRACKER                 MOD-BUFFER-CHUNK GOAL
```

If there is not a chunk in the control buffer then it will output warnings to indicate that and then generate a set-buffer-chunk event to place a chunk which contains only the control slot into the control buffer:

```
#|Warning: Tracker update occurred with no chunk in the control buffer GOAL. |#
#|Warning: Creating a new chunk with only the VALUE slot. |#
     4.038   TRACKER                 SET-BUFFER-CHUNK GOAL CHUNK2
```

In that case it will also create an event which is not displayed in the trace to remove the temporary chunk which was created to be placed into the buffer. That event would look like this if it were printed:

```
4.038   TRACKER                Clean-up unneeded chunk
```

All requests are marked as completed at the time they are made.

Modification Requests

tracker-modification

{**good-slot** *good-slot*}
{**bad-slot** *bad-slot*}
{**good-weight** *good-weight*}
{**good-scale** *good-scale*}
{**bad-weight** *bad-weight*}
{**bad-scale** *bad-scale*}
{**min** *minimum-control-value*}
{**max** *maximum-control-value*}
{**control-count** *number-of-control-values*}
{**control-scale** *control-scale*}
{**temp** *initial-temperature*}
{**scale** *temperature-scale*}
{**delay** *update-delay*}

A modification request to the tracker buffer will update the information of the tracker which is currently in the tracker buffer. The control slot may not be adjusted and neither can the buffers in which the good and bad slots are located, but all other tracker values may be changed, with the constraint that there still must be either a good or bad slot in the tracker.

After the changes are made to the tracker the tracker buffer is erased (since that chunk no longer reflects the current values for that tracker) and an immediate update of the tracker occurs.

All modification requests are marked as completed at the time they are made.

Commands

print-tracker-stats

Syntax:

print-tracker-stats {*control-slot*} -> [equation | ((control equation)*) | **nil**]
print-tracker-stats-fct {*control-slot*} -> [equation | ((control equation)*) | **nil**]

Remote command name:

print-tracker-stats

Arguments and Values:

control-slot ::= the name of a control slot which is being tracked
equation ::= (c b a)
a,b,c ::= the coefficients in the currently estimated equation for that tracker: $c + bx + ax^2$
control ::= the name of a control slot

Description:

Print-tracker-stats can be used to output the details of a tracker or all trackers to the command output trace. If a control-slot is specified and names a slot which has a tracker associated with it in the current model then the details of that tracker are printed and the coefficients of the current estimated value equation are returned. If the name given does not name a slot for which there is a tracker in the current model a warning is printed and **nil** is returned. If no control-slot is specified then the details of all the currently created trackers in the current model are printed and a list of 2-element lists is returned where the first element of a each sublist is the name of the control slot which is being tracked and the second element is the list of coefficients for that tracker's equation. If there is no current model then a warning is printed and **nil** is returned.

Examples:

```
1> (load-act-r-model "ACT-R:extras;tracker;simple.lisp")
T

2> (print-tracker-stats)
NIL

3> (run-test 50)
...
T
"response"

4> (print-tracker-stats-fct 'value)
Tracker for slot VALUE
 Good slot is GOOD in buffer IMAGINAL with weight 1
 Bad slot is BAD in buffer IMAGINAL with weight -1
  Current temperature is: 0.8395405
  Current equation is: -0.593197 + 0.7909202x + -0.19772993x^2
```

```
  Choices: 1.000 2.000 3.000
  Probs:   0.306 0.388 0.306
  Current setting is: 1

5> (module-request 'tracker (define-chunk-spec control-slot size good-slot color))
...
T

6> (print-tracker-stats)
Tracker for slot SIZE
 Good slot is COLOR in buffer IMAGINAL with weight 1
  Current temperature is: 1.0
  Current equation is: 0 + 0x + 0x^2
  Choices: 0.000 0.050 0.100 0.150 0.200 0.250 ... 0.800 0.850 0.900 0.950 1.000
  Probs:   0.048 0.048 0.048 0.048 0.048 0.048 ... 0.048 0.048 0.048 0.048 0.048
  Current setting is: 4/5
Tracker for slot VALUE
 Good slot is GOOD in buffer IMAGINAL with weight 1
 Bad slot is BAD in buffer IMAGINAL with weight -1
  Current temperature is: 0.8395405
  Current equation is: -0.593197 + 0.7909202x + -0.19772993x^2
  Choices: 1.000 2.000 3.000
  Probs:   0.306 0.388 0.306
  Current setting is: 1 ((SIZE (0 0 0)) (VALUE (-0.19844745 0.2977315 -0.099264)))

E> (print-tracker-stats-fct 'bad)
#|Warning: There is no tracker associated with the BAD slot. |#
NIL

E> (print-tracker-stats)
#|Warning: No current model available for printing tracker stats. |#
NIL
```