# C>H>R>I>S

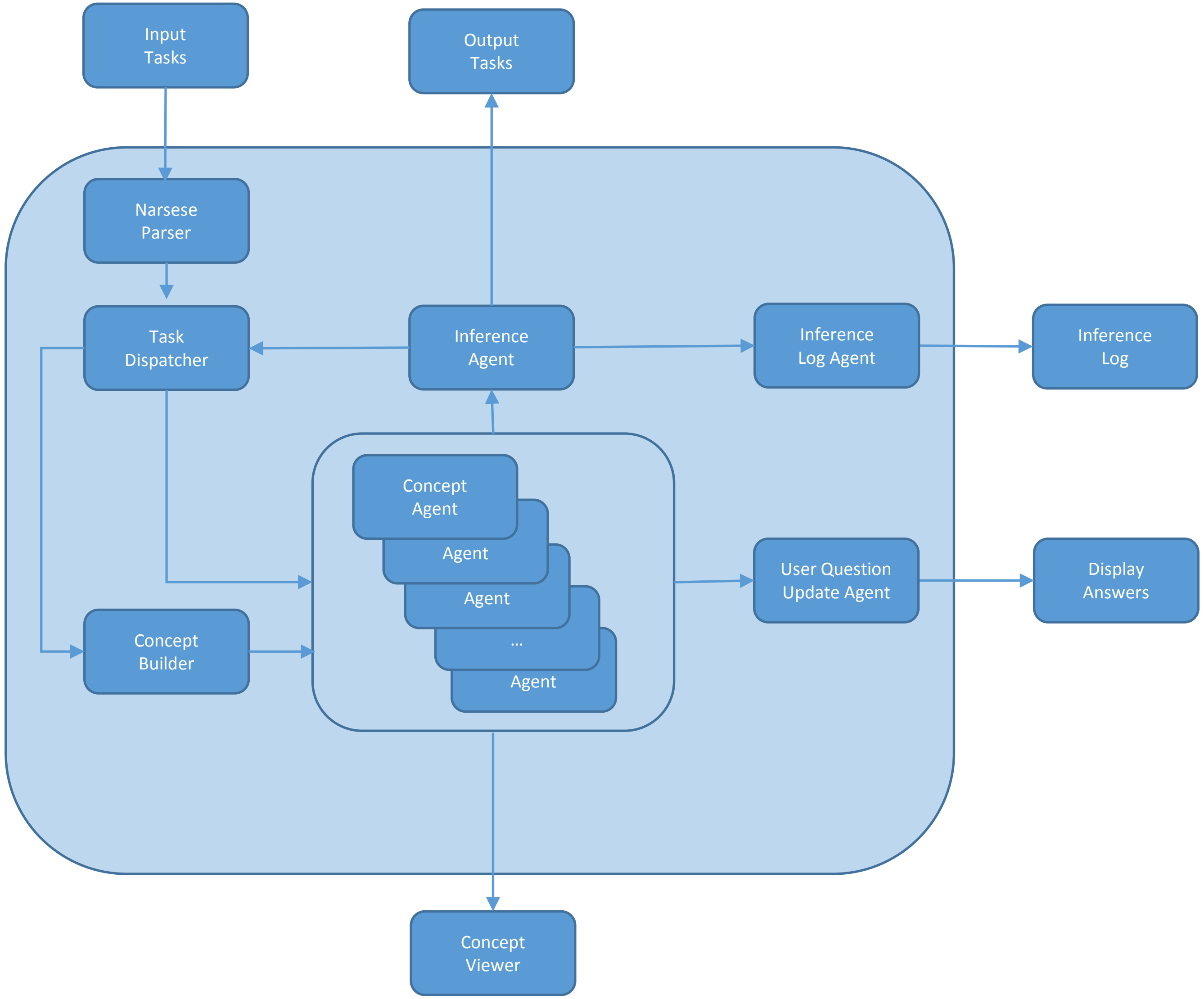
## A prototype AGI – UI User manual

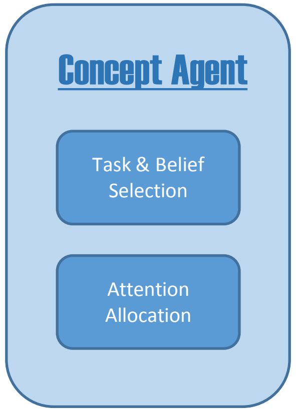
# C>H>R>I>S Features Overview

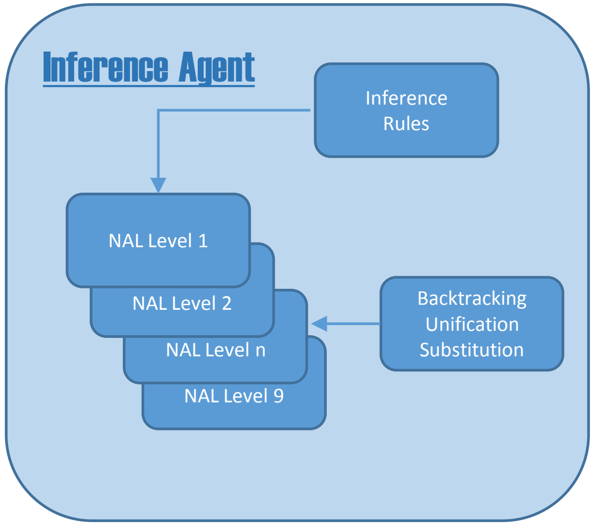
Computerised Humanlike Rationally Intelligent System aka C>H>R>I>S

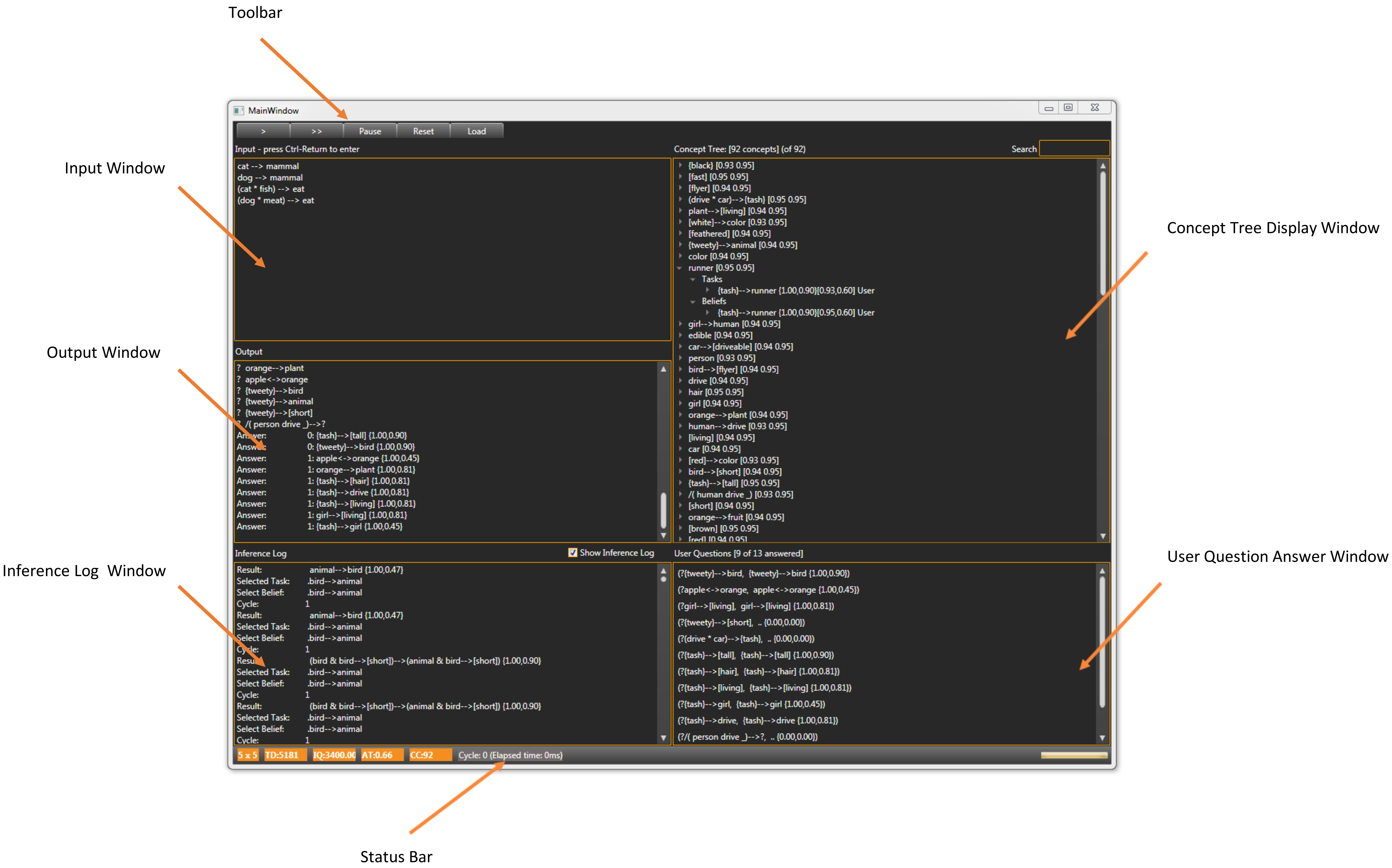
Agent based architecture with message to support multi processing environment
Every concept is an agent and can be processed in parallel
Automatically adjusts workload to maximise system resources with a dynamic activation threshold
Employs a ‘Friendly AGI’ supporting attention allocation algorithm
Utilises neural net like activation spreading incorporating activation, inhibition, decay and latency
Utilises Pei Wang’s Non Axiomatic Logic (Currently implemented to level 6)
NAL Logic levels are implemented independently and can be enabled or disabled as required
Provides an updated Grammar that includes support for constants of string, float and integer
All system parameters can be modified in XML and do not require a recompile

# C>H>R>I>S Architecture Overview









Input - press Ctrl-Return to enter

```
cat --> mammal
dog --> mammal
(cat * fish) --> eat
(dog * meat) --> eat
```

Output

```
? orange-->plant
? apple<->orange
? {tweety}-->bird
? {tweety}-->animal
? {tweety}-->[short]
? /( person drive _)-->?
Answer:         0: {tash}-->[tall] {1.00,0.90}
Answer:         0: {tweety}-->bird {1.00,0.90}
Answer:         1: apple<->orange {1.00,0.45}
Answer:         1: orange-->plant {1.00,0.81}
Answer:         1: {tash}-->[hair] {1.00,0.81}
Answer:         1: {tash}-->drive {1.00,0.81}
Answer:         1: {tash}-->[living] {1.00,0.81}
Answer:         1: girl-->[living] {1.00,0.81}
Answer:         1: {tash}-->girl {1.00,0.45}
```

Inference Log

☑ Show Inference Log

```
Result:          animal-->bird {1.00,0.47}
Selected Task:  .bird-->animal
Select Belief:  .bird-->animal
Cycle:          1
Result:          animal-->bird {1.00,0.47}
Selected Task:  .bird-->animal
Select Belief:  .bird-->animal
Cycle:          1
Result:          (bird & bird-->[short])-->(animal & bird-->[short]) {1.00,0.90}
Selected Task:  .bird-->animal
Select Belief:  .bird-->animal
Cycle:          1
Result:          (bird & bird-->[short])-->(animal & bird-->[short]) {1.00,0.90}
Selected Task:  .bird-->animal
Select Belief:  .bird-->animal
Cycle:          1
```

Concept Tree: [92 concepts] (of 92)

```
{black} [0.93 0.95]
[fast] [0.95 0.95]
[flyer] [0.94 0.95]
(drive * car)-->{tash} [0.95 0.95]
plant-->[living] [0.94 0.95]
[white]-->color [0.93 0.95]
[feathered] [0.94 0.95]
{tweety}-->animal [0.94 0.95]
color [0.94 0.95]
runner [0.95 0.95]
  Tasks
    {tash}-->runner {1.00,0.90}[0.93,0.60] User
  Beliefs
    {tash}-->runner {1.00,0.90}[0.95,0.60] User
girl-->human [0.94 0.95]
edible [0.94 0.95]
car-->[driveable] [0.94 0.95]
person [0.93 0.95]
bird-->[flyer] [0.94 0.95]
drive [0.94 0.95]
hair [0.95 0.95]
girl [0.94 0.95]
orange-->plant [0.94 0.95]
human-->drive [0.93 0.95]
[living] [0.94 0.95]
car [0.94 0.95]
[red]-->color [0.93 0.95]
bird-->[short] [0.94 0.95]
{tash}-->[tall] [0.95 0.95]
/( human drive _) [0.93 0.95]
[short] [0.94 0.95]
orange-->fruit [0.94 0.95]
[brown] [0.95 0.95]
[red] [0.94 0.95]
```

User Questions [9 of 13 answered]

```
(?{tweety}-->bird,  {tweety}-->bird {1.00,0.90})
(?apple<->orange,  apple<->orange {1.00,0.45})
(?girl-->[living],  girl-->[living] {1.00,0.81})
(?{tweety}-->[short],  .. {0.00,0.00})
(?(drive * car)-->{tash},  .. {0.00,0.00})
(?{tash}-->[tall],  {tash}-->[tall] {1.00,0.90})
(?{tash}-->[hair],  {tash}-->[hair] {1.00,0.81})
(?{tash}-->[living],  {tash}-->[living] {1.00,0.81})
(?{tash}-->girl,  {tash}-->girl {1.00,0.45})
(?{tash}-->drive,  {tash}-->drive {1.00,0.81})
(?/( person drive _)-->?,  .. {0.00,0.00})
```

```
5 x 5   TD:5181   IQ:3400.00   AT:0.66   CC:92   Cycle: 0 (Elapsed time: 0ms)
```

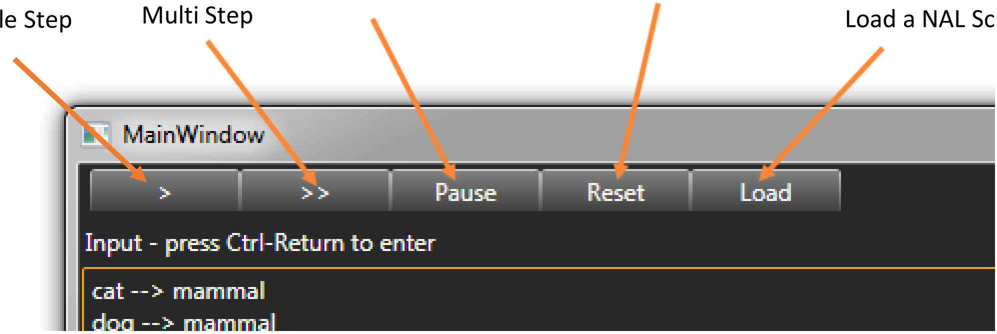
Single Step

Multi Step

Pause

Reset Memory

Load a NAL Script file

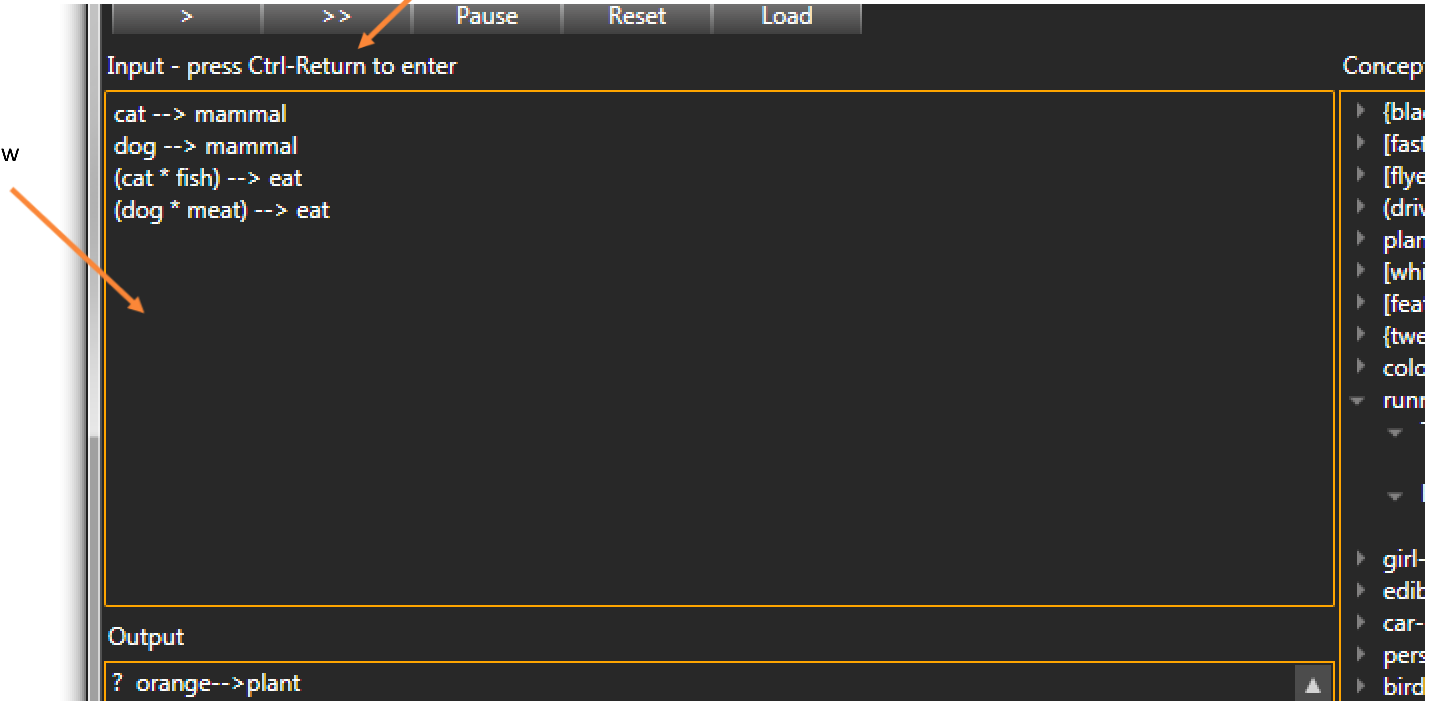


Input - press Ctrl-Return to enter

```
cat --> mammal
dog --> mammal
```

Use Ctrl + Enter to input text

NAL Sentence input window

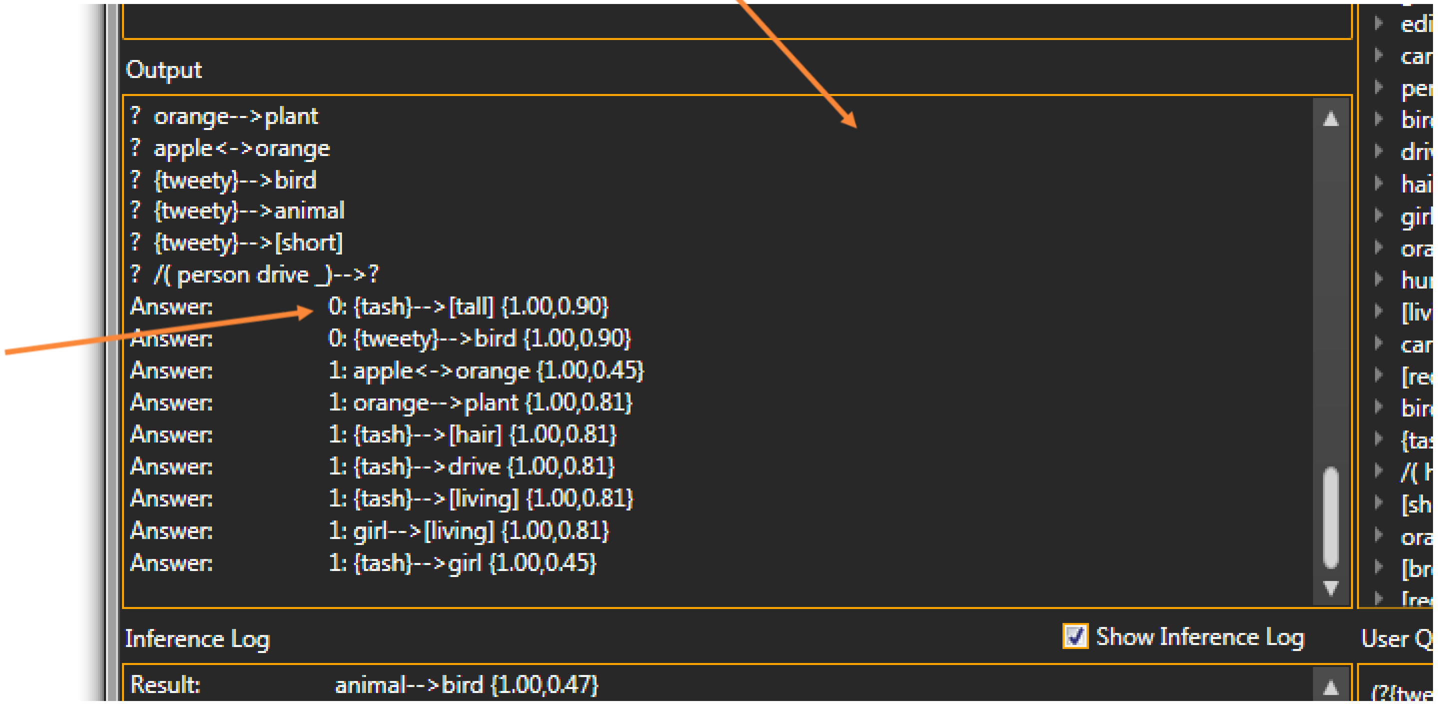


```
cat --> mammal
dog --> mammal
(cat * fish) --> eat
(dog * meat) --> eat
```

```
? orange-->plant
```

NAL Output window

Answer to User Question
with system cycle



```
?  orange-->plant
?  apple<->orange
?  {tweety}-->bird
?  {tweety}-->animal
?  {tweety}-->[short]
?  /( person drive _)-->?
Answer:            0: {tash}-->[tall] {1.00,0.90}
Answer:            0: {tweety}-->bird {1.00,0.90}
Answer:            1: apple<->orange {1.00,0.45}
Answer:            1: orange-->plant {1.00,0.81}
Answer:            1: {tash}-->[hair] {1.00,0.81}
Answer:            1: {tash}-->drive {1.00,0.81}
Answer:            1: {tash}-->[living] {1.00,0.81}
Answer:            1: girl-->[living] {1.00,0.81}
Answer:            1: {tash}-->girl {1.00,0.45}
```

```
Result:            animal-->bird {1.00,0.47}
```

# C>H>R>I>S Grammar

```
 task                   ::==     [attention] sentence
sentence                ::==     judgement | question | goal
judgement               ::==     [tense-operator] statement [truth]
goal                    ::==     ‘!’ statement [desire]
question                ::==     {‘?’ | '??'} [tense-operator] statement
statement               ::==     term [infix-operator  term] | term binary-operator term
term                    ::==     constant | variable | set | ‘(‘ statement ‘)’ | '--'  '(' term ')' | prefix-operator '(' term {term}+ ')'
set                     ::==     '{' {term}+ '}' | '[' {term}+ ']'
infix-operator          ::==     '&&' | '||' ',' | ';' | '&' | '|'| '*'
copula                  ::==     '-->' | '<->'  | '{--' | '--]' | '{-]'  '==>' | '<=>' | '/=>' | '\=>' | '|=>' | '</>' | '<|>'
binary-operator         ::==     '-' | '~' | copula
prefix-operator         ::==     '\' | '/' | '^'
tense-operator          ::==     ':/:' | ':|:' | ':\:'
variable                ::==     independent-variable | dependent-variable | query-variable
independent-variable    ::==     ‘#’ constant
dependent-variable      ::==     ‘$’ constant
query-variable          ::==     ‘?’ [constant]
constant                ::==     string-literal | decimal-integer | real-number
string-literal          ::==     letter {letter | digit | '_'}
decimal-integer         ::==     ['-' | '+'] digit-sequence
digit-sequence          ::==     digit {digit}
real-number             ::==     ['-' | '+'] digit-sequence '.' digit-sequence
truth                   ::==     floatTuple
desire                  ::==     floatTuple
attention               ::==     floatTuple
floatTuple              ::==     '{' real-number  real-number '}'
```

# C>H>R>I>S Grammar Examples

NAL 1

| | |
|---|---|
| cat --> animal | white space is optional |
| cat --> animal {1.0 0.9} | with optional truth value: note that there are no separating commas |
| ? cat --> ? | Question marks go at the head of the sentence |

NAL 2

| | |
|---|---|
| {Tweety} --> canary | sentences do not need full stops – this is the default |
| canary --> [fly] | sentences do not need to be enclosed in ‘<‘ or ‘>’ |
| orange <-> apple | |

NAL 3

| | |
|---|---|
| {orange apple pear} -> fruit | terms should not be separated with commas |
| (black & board) --> blackboard | |
| robin --> (mammal - swimmer) {0.00  0.9} | ’-’ cannot be used in constant names, use ’_’ instead |

NAL 4

| | |
|---|---|
| (acid * base) --> reaction | operators are generally infix |
| \(neutralization _ base) --> acid {0.80 0.90} | images are an exception, they are prefix |

NAL 5

| | |
|---|---|
| (robin --> [flying]) ==> (robin --> bird) | use parenthesis to distinguish statements in higher order statements |

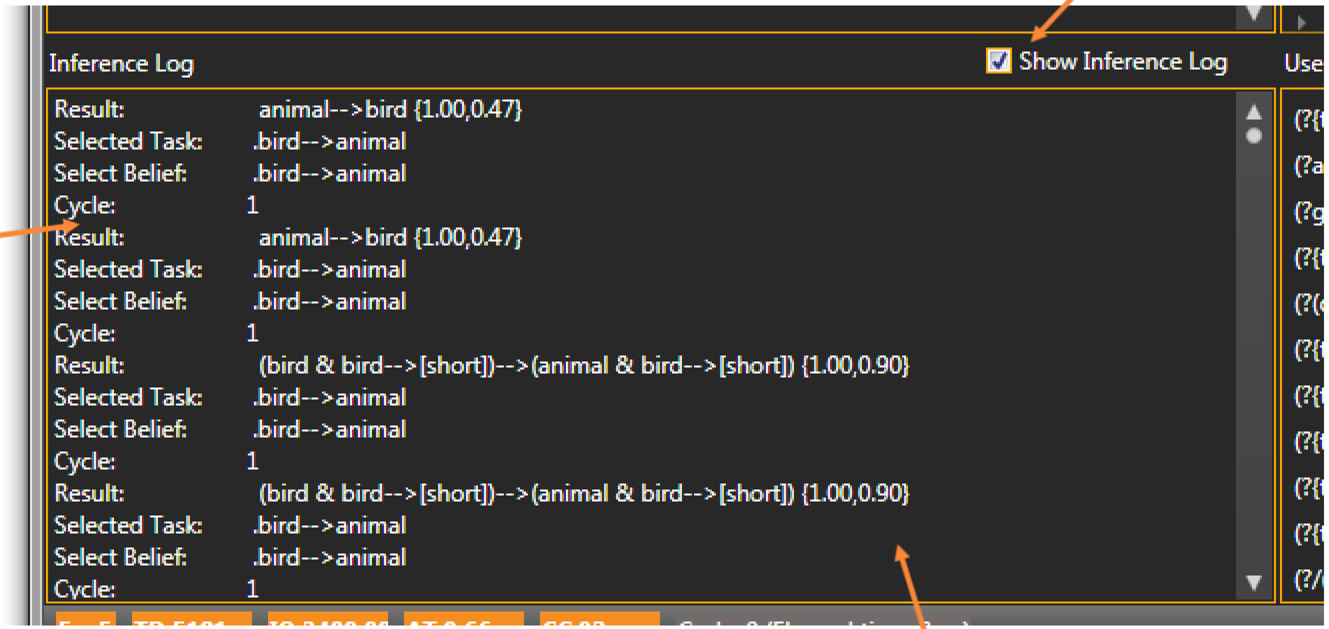
NAL 6

($1 --> lock) ==> ((#2 --> key) && ($1 --> /(open #2 _)))

{key1}-->key

? ($1 --> lock) ==> (({key1} --> key) && ($1 --> /(open {key1} _)))

Inference Log Enable Checkbox.
Window is only updated when
single stepping.

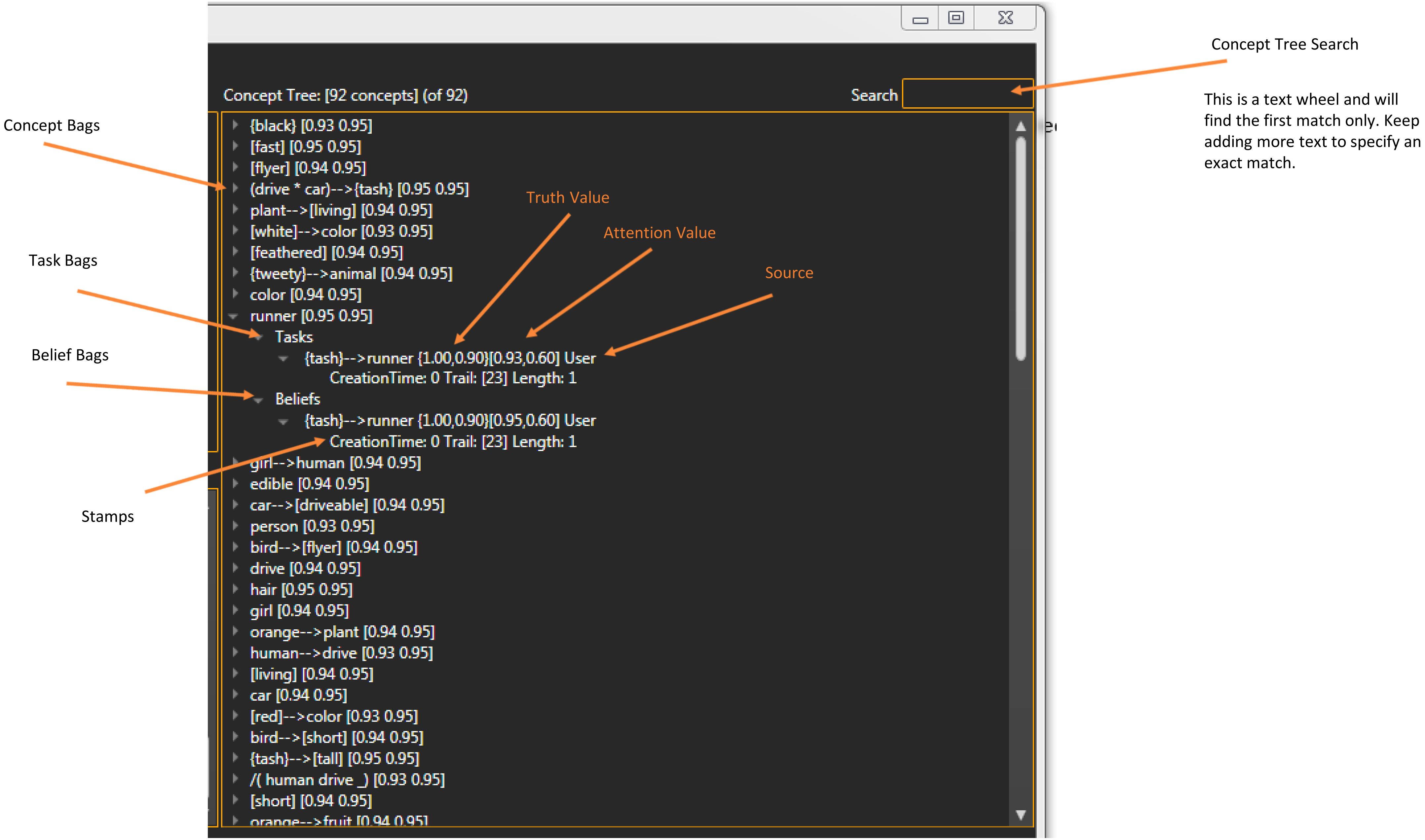


```
Result:           animal-->bird {1.00,0.47}
Selected Task:   .bird-->animal
Select Belief:   .bird-->animal
Cycle:           1
Result:           animal-->bird {1.00,0.47}
Selected Task:   .bird-->animal
Select Belief:   .bird-->animal
Cycle:           1
Result:           (bird & bird-->[short])-->(animal & bird-->[short]) {1.00,0.90}
Selected Task:   .bird-->animal
Select Belief:   .bird-->animal
Cycle:           1
Result:           (bird & bird-->[short])-->(animal & bird-->[short]) {1.00,0.90}
Selected Task:   .bird-->animal
Select Belief:   .bird-->animal
Cycle:           1
```

Inference Results

There can be a lot of results per cycle. Multiple concepts are activated per cycle and each concept can have multiple tasks and beliefs selected for inference. This is dependant on the system parameters.

Inference Log Window



```
Concept Tree: [92 concepts] (of 92)                                  Search
{black} [0.93 0.95]
[fast] [0.95 0.95]
[flyer] [0.94 0.95]
(drive * car)-->{tash} [0.95 0.95]
plant-->[living] [0.94 0.95]
[white]-->color [0.93 0.95]
[feathered] [0.94 0.95]
{tweety}-->animal [0.94 0.95]
color [0.94 0.95]
runner [0.95 0.95]
  Tasks
    {tash}-->runner {1.00,0.90}[0.93,0.60] User
      CreationTime: 0 Trail: [23] Length: 1
  Beliefs
    {tash}-->runner {1.00,0.90}[0.95,0.60] User
      CreationTime: 0 Trail: [23] Length: 1
girl-->human [0.94 0.95]
edible [0.94 0.95]
car-->[driveable] [0.94 0.95]
person [0.93 0.95]
bird-->[flyer] [0.94 0.95]
drive [0.94 0.95]
hair [0.95 0.95]
girl [0.94 0.95]
orange-->plant [0.94 0.95]
human-->drive [0.93 0.95]
[living] [0.94 0.95]
car [0.94 0.95]
[red]-->color [0.93 0.95]
bird-->[short] [0.94 0.95]
{tash}-->[tall] [0.95 0.95]
/( human drive _) [0.93 0.95]
[short] [0.94 0.95]
orange-->fruit [0.94 0.95]
```

Concept Tree Search

This is a text wheel and will find the first match only. Keep adding more text to specify an exact match.

Concept Bags

Task Bags

Belief Bags

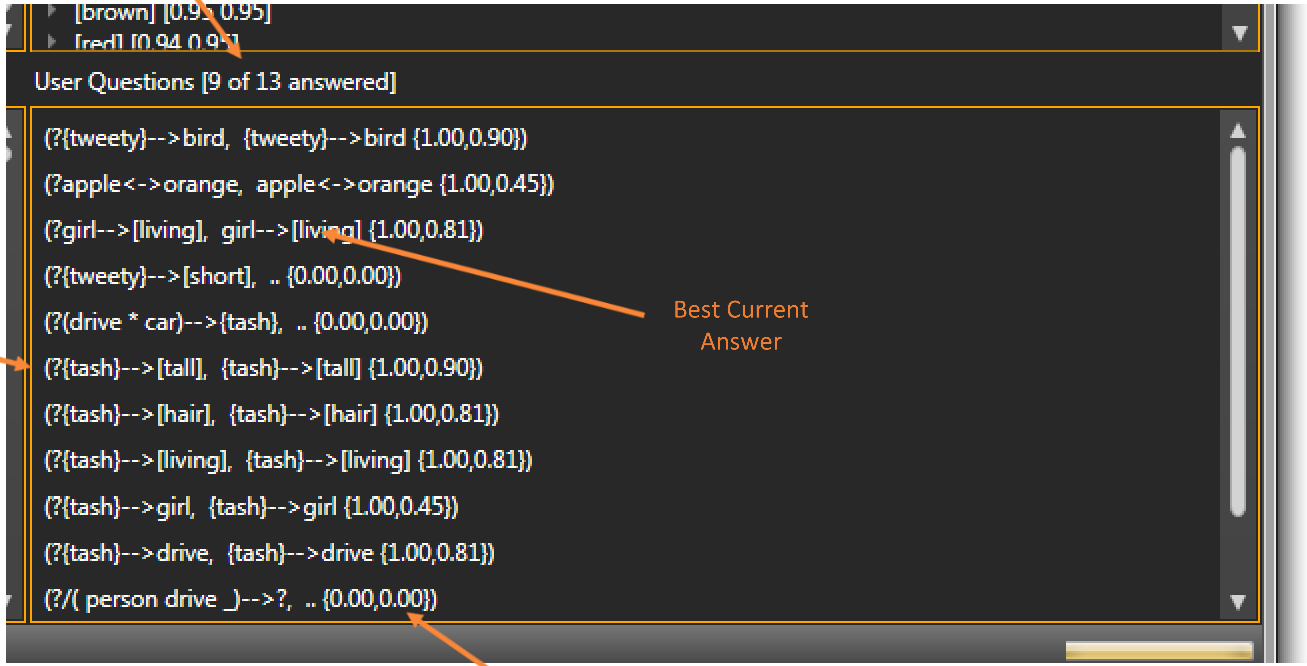
Stamps

Truth Value

Attention Value

Source

Number of Current User Questions that have been answered

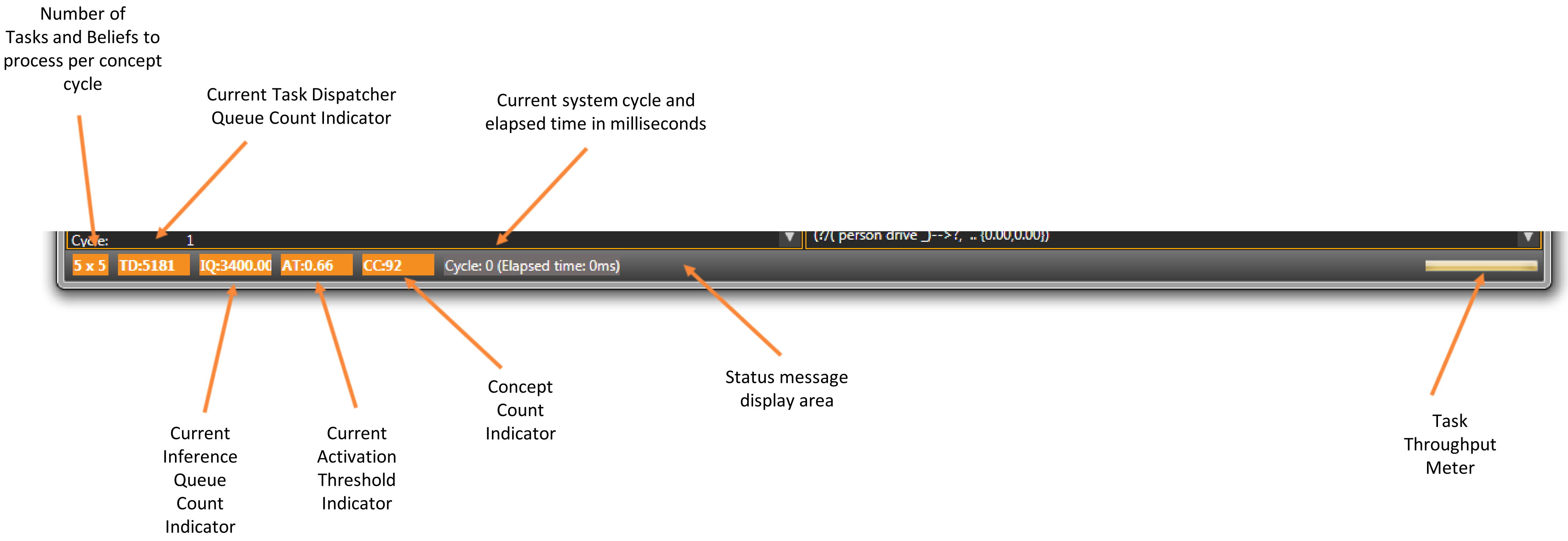


```
[brown] [0.95 0.95]
[red] [0.94 0.95]
User Questions [9 of 13 answered]
(?{tweety}-->bird,  {tweety}-->bird {1.00,0.90})
(?apple<->orange,  apple<->orange {1.00,0.45})
(?girl-->[living],  girl-->[living] {1.00,0.81})
(?{tweety}-->[short],  .. {0.00,0.00})
(?(drive * car)-->{tash},  .. {0.00,0.00})
(?{tash}-->[tall],  {tash}-->[tall] {1.00,0.90})
(?{tash}-->[hair],  {tash}-->[hair] {1.00,0.81})
(?{tash}-->[living],  {tash}-->[living] {1.00,0.81})
(?{tash}-->girl,  {tash}-->girl {1.00,0.45})
(?{tash}-->drive,  {tash}-->drive {1.00,0.81})
(?/( person drive _)-->?,  .. {0.00,0.00})
```

Questions to be answered

Best Current Answer

Best Current Truth Value



Cycle: 1

{?/( person drive _)-->?, .. {0.00,0.00}}

5 x 5 TD:5181 IQ:3400.00 AT:0.66 CC:92 Cycle: 0 (Elapsed time: 0ms)

Number of Tasks and Beliefs to process per concept cycle

Current Task Dispatcher Queue Count Indicator

Current system cycle and elapsed time in milliseconds

Current Inference Queue Count Indicator

Current Activation Threshold Indicator

Concept Count Indicator

Status message display area

Task Throughput Meter

# C>H>R>I>S System Parameters

| | |
|---|---|
| CYCLES | Max system cycles - set to 0L for infinite cycles |
| LATENCY_PERIOD | Concept latency period in system cycles |
| MS_PER_CYCLE | Milliseconds per cycle (77 cycles = 12 hz human alpha wave e.g. 1000 / 12) |
| ACTIVATION_THRESHOLD | Initial Concept Activation Threshold - when threshold is above priority concept is activated |
| INFERENCE_TASKS_PER_CYCLE | Number of tasks to select for each inference task (per concept) |
| INFERENCE_BELIEFS_PER_CYCLE | Number of beliefs to select for each inference task (per concept) |
| CONFIDENCE | Truth Value confidence component |
| FREQUENCY | Truth Value frequency component |
| MINIMUM_CONFIDENCE | don't accept inference results with confidence below this Value |
| STI | Short Term Importance default Value AKA priority |
| LTI | long Term Importance default Value AKA duration |
| USERSTI | Short Term Importance default Value for user entered Values AKA priority |
| USERLTI | long Term Importance default Value for user entered Values AKA duration |
| TRAIL_LENGTH | maximum length allowed for inference trail within stamp |
| CONCEPT_SELECTION_FACTOR | Determines the attentional focus - the > the Value the < the selection range |
| TASK_SELECTION_FACTOR | Determines the attentional focus - the > the Value the < the selection range |
| BELIEF_SELECTION_FACTOR | Determines the attentional focus - the > the Value the < the selection range |
| NOVELTASK_SELECTION_FACTOR | Determines the attentional focus - the > the Value the < the selection range |
| RAZOR_PARAMETER | Syntactic simplicity adjustment parameter (e/n^r) |
| DECISION_THRESHOLD | Accepts goals above this threshold |
| CONCEPT_CAPACITY | Size of concept pool in Working memory |
| BELIEF_CAPACITY | Size of Belief pool within each concept |
| TASK_CAPACITY | Size of Task pool within each concept |
| INFERENCE_THREADS | Number of threads to run the inference step - one concept per thread (Deprecated) |
| STATUS_UPDATE_PERIOD | Number of cycles to wait before updating the status bar |
| NEW_TASKS_PER_THREAD_MAX | Maximum number of new tasks per thread (Deprecated) |
| NEW_TASKS_SYSTEM_MAX | Maximum number of new tasks per system |
| NOVEL_TASK_EXPECTATION_THRESHOLD | Threshold above which new tasks are accepted as being novel |
| INFERENCE_SEARCH_DEPTH | Adjusts decay rate of satisfied tasks - 1.0 to inf where 1.0 is low decay and higher is greater |
| DECAY_RATE | Decay rate tuning parameter - higher = slower decay rate (used in Attention.Decay) |
| TASKBAG_INSERTION_THRESHOLD | Minimium Priority value for insertion into task bag |
| BELIEFBAG_INSERTION_THRESHOLD | Minimium Priority value for insertion into belief bag |

Parameters are stored in XML in the DefaultParameters files

# C>H>R>I>S Test Scripts

There are a range of scripts available to demonstrate the use of the modified grammar:-

NALLevel1.txt
NALLevel2.txt
NALLevel3.txt
NALLevel4.txt
NALLevel5.txt
NALLevel6.txt
NALLevel7.txt
NALLevel8.txt
NALLevel9.txt

Note that these scripts do not provide the same capability as ‘OpenNARS’ to ‘Reset’ the memory during execution of a script. To test individual rules you need to manually enter the tests (or cut and paste).

However, CHRIS is much better than OpenNARS at managing multiple user inputs tasks. You can enter a script with 100’s of tasks and it will ‘generally’ find most of the solutions.

To supplement the above scripts there are additional files that provide more challenging tasks, such as multi step inference, deduction chains and simple logic tasks.

Fruit example.txt
John knows.txt
Tweety plane problem.txt
NAL5 conditional.txt
Deduction chain.txt
Robot problem.txt
Family tree.txt
Test script.txt
Combined Test.txt

The combined Test above incorporates all the other tests into a single file and is good for stress testing.