

Relatório 3 - IA941 – 2017

Controlando o WorldServer3D

CLARION

Fabio Mariotto de Azevedo – RA 208985 - mariotto.pk@gmail.com -15/05/2017

1 SUMÁRIO

2	Introdução	3
3	Arquitetura CLARION.....	4
4	Arquitetura de controle do WS3D.....	6
4.1	Definição do problema	6
4.2	Abordagem ao problema	6
4.3	Os sentidos da criatura.....	7
4.4	Possíveis ações	8
4.5	Objetivos e motivações	8
4.6	Reforço positivo	9
4.7	Parâmetros de Aprendizagem.....	10
4.8	Interface	10
4.9	Registro do aprendizado	11
5	Resultados	12
6	Próximos passos	13
7	Conclusão	13

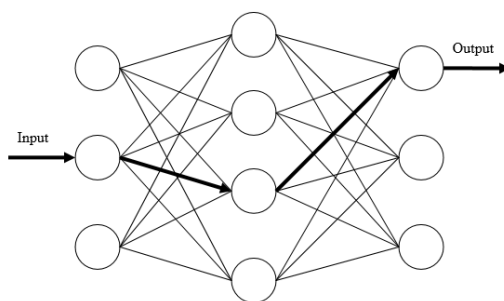
2 INTRODUÇÃO

O objetivo desse relatório é apresentar os resultados obtidos utilizando-se a arquitetura cognitiva CLARION (*Connectionist Learning with Adaptive Rule Induction ON-Line*) para controlar uma criatura no ambiente virtual WorldServer3D. O objetivo da criatura é encontrar 3 joias listadas aleatoriamente. A energia da criatura diminui com o passar do tempo e a mesma deve procurar comida para se manter viva.

3 ARQUITETURA CLARION

A arquitetura cognitiva CLARION (*Connectionist Learning with Adaptive Rule Induction ON-Line*) desenvolvida pelo Dr. Ron Sun tem como objetivo ser funcionalmente similar a cognição humana.

A simulação da representação de conhecimento no CLARION é dividida em conhecimento explícito e implícito. Por exemplo, conhecimento implícito não é diretamente acessível, mas pode ser usado para processamento. Uma "backpropagation neural network" é utilizada para representar esse conhecimento implícito. Essa rede neural é baseada na biologia do cérebro humano. Nessa rede, um sinal atravessa um caminho onde os nós representam neurônios e as ligações representam sinapses. Os caminhos pelos quais cada sinapse impacta um nó é calculado pela propagação reversa, verificando o nível de reforço positivo em relação aos inputs dados.



No exemplo acima um caminho específico foi treinado de forma que quando um input é ativado, o sinal atravessa através da rede para o output treinado. Assim como neurônios, os nós não são funcionais individualmente. Apenas quando são parte de uma rede que seus comportamentos são construtivos.

O conhecimento explícito, por outro lado, é representado mais apropriadamente de uma forma simbólica e local, de forma que apresentam mais sentido mesmo individualmente. Dessa forma o conhecimento explícito é composto por partes de informação que podem diretamente ser acessível e manipuladas. Enquanto conhecimento implícito representa informação subconsciente, o conhecimento explícito representa informação consciente.

Dentro de cada nível da arquitetura existem vários módulos que lidam com as funções cognitivas. Esses módulos fazem partes de subsistemas. Por exemplo, o ACS (action-centered subsystem) que contém os processos que governam todas as ações do modelo cognitivo. Ele é composto por módulos que existem em ambos os níveis, implícito e explícito. O subsistema NACS (non-action-centered subsystem) também contém módulos em ambos os níveis. Em adição a esses subsistemas também existem os sistemas motivacional e meta-cognitivo. O sistema motivacional é responsável por guiar o comportamento geral da criatura, através da formação de "goals" (objetivos) que são então enviados a uma estrutura de objetivos.

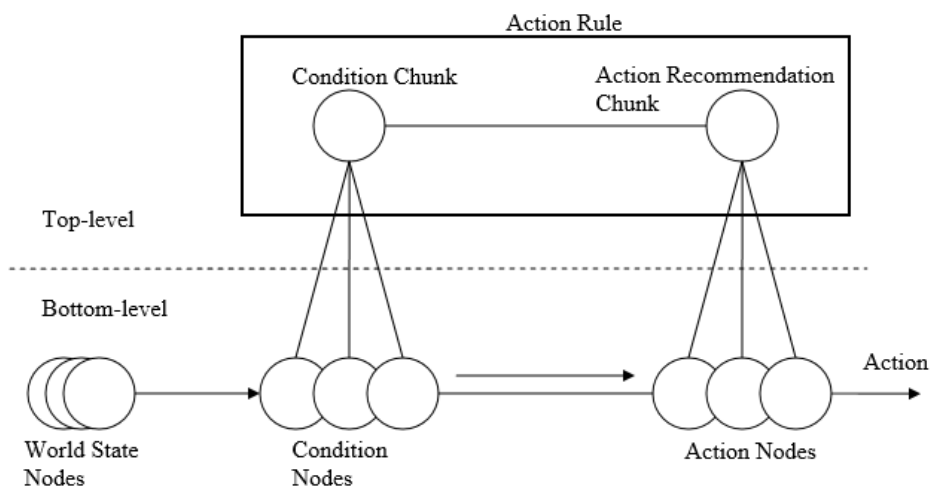
		Sub-system	
		Action-centered	Non-action-centered
Knowledge	Top-level (Explicit)	Module(s)	Module(s)
	Bottom-level (Implicit)	Module(s)	Module(s)

Os “inputs” (entradas) de informação do ambiente são feitos através de pares “dimension/value” que descrevem como e quais objetos estão no ambiente. Esses pares em conjunto com a memória e os objetivos constituem o estado corrente do universo, que é então utilizado como input pelo ACS.

Para conhecimento explícito são utilizados os chunks e rules. Eles existem no nível explícito da arquitetura. Chunks servem como uma forma de amarrar os pares dimension/value em grupos de forma que eles descrevam algo em particular. Regras são usadas pelo conhecimento explícito e diferem suavemente baseado em qual subsistemas eles estão. Para módulos do ACS, elas são regras de ação. Se forem do NACS elas são regras de associação. Assim, o conhecimento explícito no CLARION é dividido em “rules” (regras) e “chunks” (nacos).

As regras de ação terão pares dimension/value como condição, e se essas estiverem presentes no ambiente, a ação relativa a regra será executada. As regras associativas também terão pares dimension/value como condição, porém sua saída é a formação de um chunk de memória.

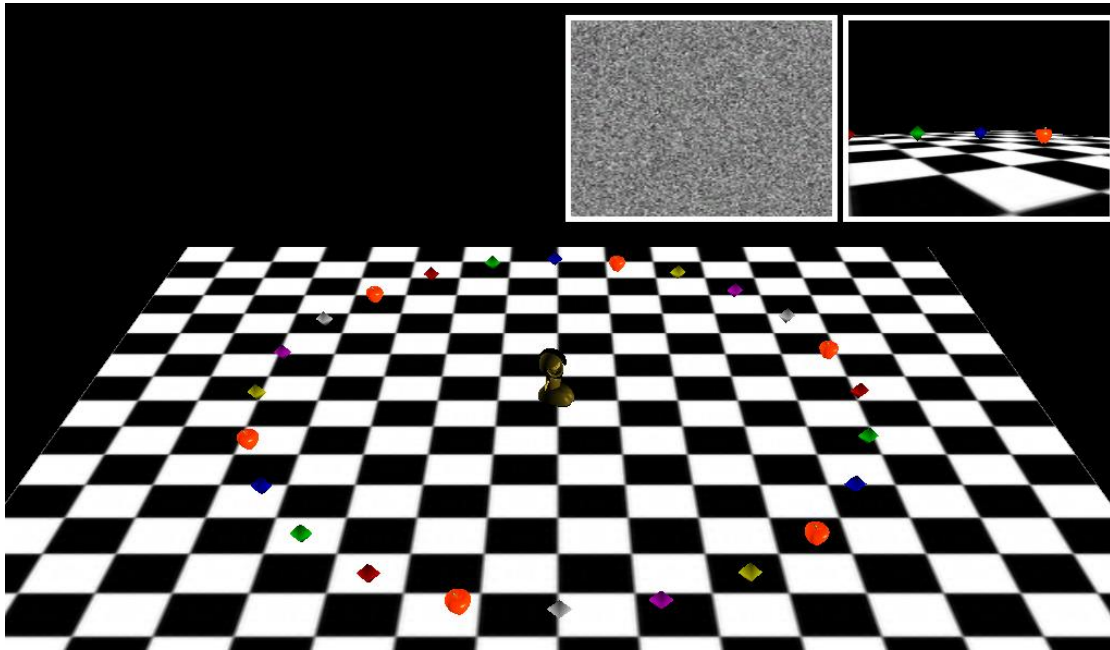
As regras são compostas em chunks e existem no nível explícito. Porém, elas são associadas com os pares dimension/value no nível implícito. Assim o nível explícito e implícito interagem formando a arquitetura de decisão.



4 ARQUITETURA DE CONTROLE DO WS3D

4.1 DEFINIÇÃO DO PROBLEMA

O desafio que a resolver é desenvolver uma mente para a criatura virtual que existe no ambiente WS3D. A criatura inicia sua vida com 1000 pontos de energia que se esvaem com o tempo, e que podem ser reabastecidos comendo uma das comidas que esteja a seu redor. A criatura deve coletar 3 joias de cores definidas aleatoriamente antes de morrer de fome.



4.2 ABORDAGEM AO PROBLEMA

O objetivo da implementação é não somente de solucionar o problema acima, mas também o de explorar ao máximo as funcionalidades da arquitetura cognitiva CLARION. Dessa forma evitara-se a utilização de regras explícitas e pré-programadas e o priorizara-se as funções que exigem a exploração e aprendizado por parte da própria criatura.

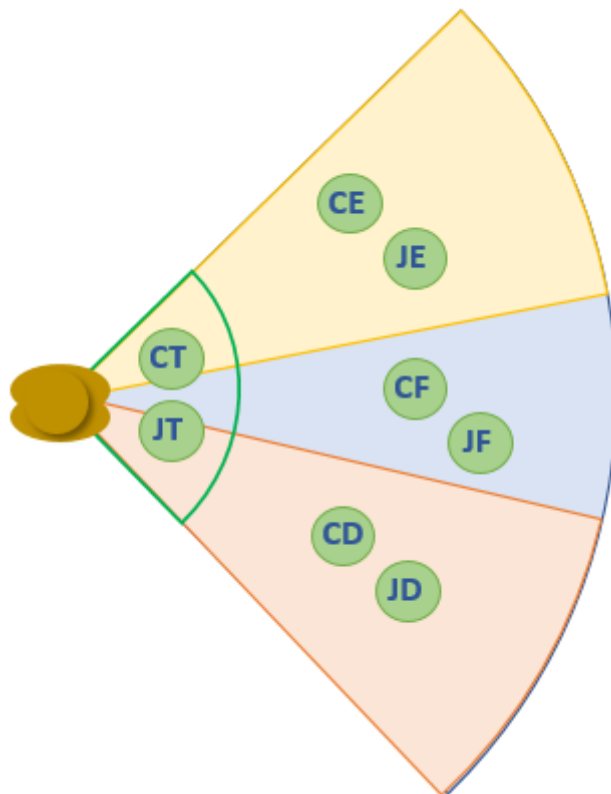
4.3 OS SENTIDOS DA CRIATURA

Para que a criatura possa captar o ambiente ao seu redor e assim ponderar suas ações, foram criados 6 sentidos, ou no caso da arquitetura, 8 “inputs”. Esses 8 inputs são os nós iniciais alimentados pela percepção da criatura ao ambiente a seu redor que serão utilizados pela rede neural da criatura, para aprender quais ações tomar.

Esses inputs são:

1. Contato com comida (CT);
2. Comida à frente (CF);
3. Comida à esquerda (CE);
4. Comida à direita (CD);
5. Contato com joia (JT);
6. Joia à frente (JF);
7. Joia à esquerda (JE);
8. Joia à direita (JD);

Eles representam a presença ou não de comida e joia (da cor necessária) no campo de visão da criatura de acordo com o esquema abaixo:

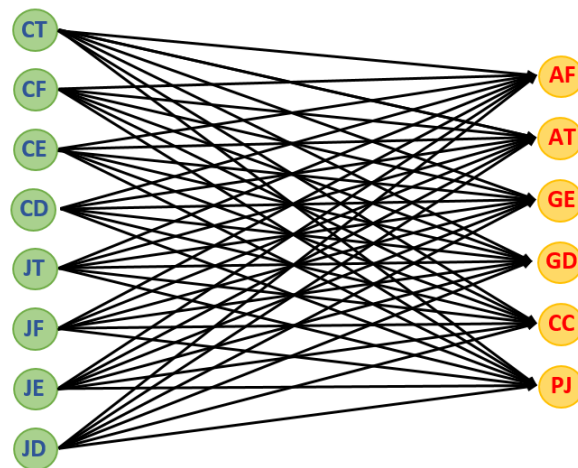


4.4 POSSÍVEIS AÇÕES

A criatura pode executar uma das seguintes ações:

1. Andar para frente (AF);
2. Andar para trás (AT);
3. Girar a esquerda (GE);
4. Girar a direita (GD);
5. Comer uma comida próxima (CC);
6. Pegar uma joia (PJ).

A arquitetura utiliza de um método conhecido como “Q-learning”, onde dados os inputs do ambiente, ações cujos resultados são positivos serão reforçadas através de um feedback positivo. Dessa forma temos algo como a rede abaixo, onde o conjunto de inputs irá ativar uma ou mais saídas:



4.5 OBJETIVOS E MOTIVAÇÕES

A criatura possui dois objetivos:

1. Procurar joias;
2. Procurar comida.

Esses objetivos funcionam como inputs que participaram da definição das regras aprendidas pela criatura. Assim, dado o objetivo atual, os inputs da criatura poderão ativar ações diferentes.

Para definir qual objetivo é priorizado pela criatura, dois “drives” concorrem pela atenção da criatura:

1. Drive por comida;
2. Drive por dominância.

O drive por dominância começa por padrão com maior relevância, porém a partir do momento que a energia da criatura se aproxima de um valor previamente definido, ela passará a optar pelo objetivo de buscar comida ao invés de buscar joias.

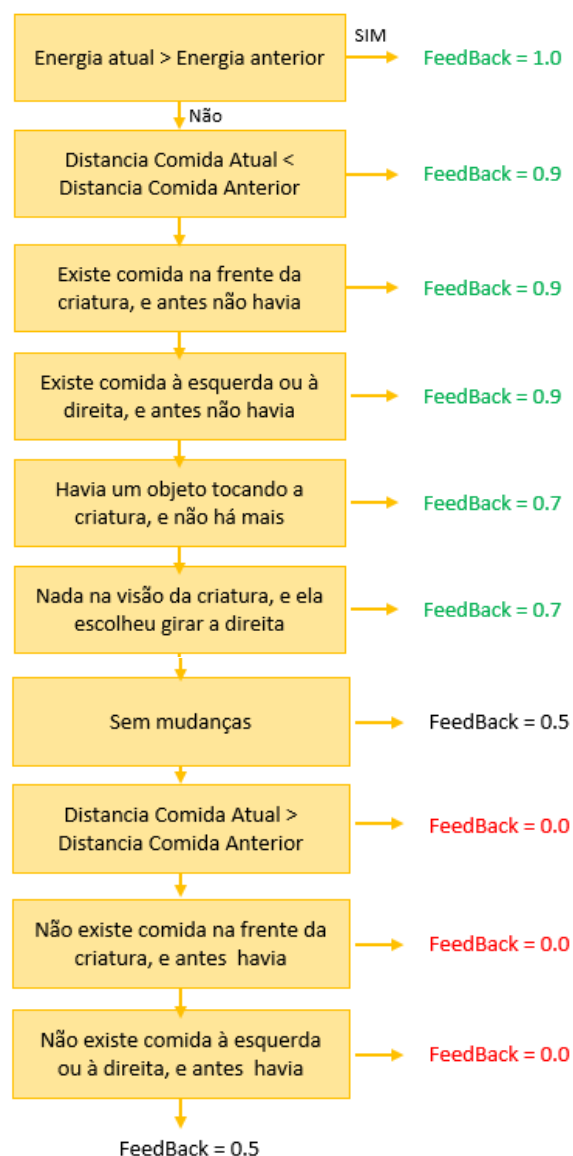
4.6 REFORÇO POSITIVO

O reforço é dado a criatura baseado nos resultados de suas ações. Para isso é gravado o estado anterior a cada ação e comparado com o seguinte. Um feedback de 0.5 é neutro. Qualquer feedback maior ou menor que esse valor será positivo ou negativo, respectivamente.

Caso o feedback seja dado considerando a ação escolhida pela criatura, estaríamos indiretamente pré-configurando seu aprendizado. Dessa forma todos os feedbacks foram configurados baseando-se apenas no status da criatura, sem nenhuma premissa.

Após testes notou-se que a criatura aprendia muito devagar nos seus primeiros ciclos. Assim, foi criada uma única regra de feedback baseada na ação tomada. Essa regra possui um feedback suavemente positivo que é dado toda vez que a criatura escolhe a ação de girar para direita quando não existe nenhuma comida ou joia no seu campo de visão.

A cadeia de decisão do feedback quando o goal selecionado é o de buscar comida:



Existem uma cadeia semelhante para quando o goal selecionado é o de buscar joia, porém no mesmo avalia-se a presença e a distância até as joias, bem como a redução do leaflet ao invés do aumento de energia.

4.7 PARÂMETROS DE APRENDIZAGEM

Existem diversos parâmetros de configuração como por exemplo especialização e generalização. Alguns desses parâmetros podem ser escolhidos durante a execução e outros foram definidos no código do programa. Todos os parâmetros relevantes são gravados e documentados nos arquivos que apresentam os resultados de execução.

4.8 INTERFACE

Durante a execução, uma interface mostrara interativamente os ciclos de cognição a criatura. Os ciclos podem ser executados de 1 em 1 apertando-se a tecla enter. Pode-se também definir uma quantidade de ciclos a ser executada antes de pausar novamente, ou então pode-se executar indefinidamente.

```
---Remaining Leaflet---
|  RED=0    GREEN=0  |
|  BLUE=0   YELLOW=0 |
|  PURP=3   WHITE=0  |
|-----|

Goal: Search Gem
C Pitch: 71
Things: (3)Yellow[82]<63>S | (3)Magenta[100]<100>S |
Gem State: Changed
Distance to Gem:FARTHER
|xX|  |xX|
|      |
FUEL: 1000
Chosen action: GO_BACK
FeedBack given: 0.5

Cycle: 7151
Creature Life: 9
Completed Leaflets: 4 (Jewels = 15)
learned Rules: 11
Running more 3852 cycles before pausing again.
```

Na interface estão apresentados, na ordem:

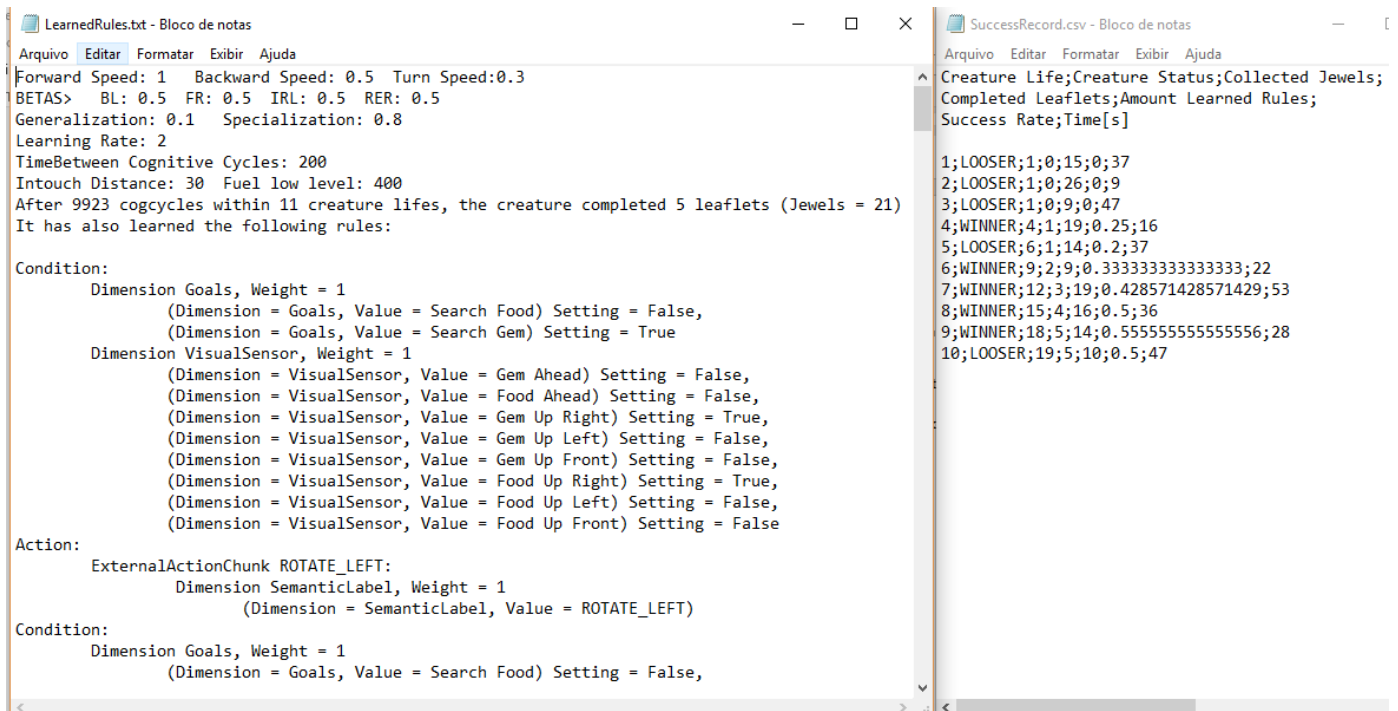
1. Número de joias pendentes a serem encontradas, por cor;
2. Goal atual da criatura;
3. Ângulo da criatura em relação ao eixo horizontal;
4. “Things” (coisas) no campo de visão da criatura, no padrão (ID)cor[distancia]<ângulo>Side/Front;
5. Indicador se os inputs da criatura alteraram após sua ação;
6. Indicador se a criatura se aproximou ou afastou de sua joia/comida;
7. Campo que mostra se existe joia (X) ou comida (O) no campo de visão frontal e lateral da criatura;
8. Campo que mostra se existe objeto tocando a criatura;
9. “Fuel” (energia) da criatura;
10. Ação escolhida pela criatura;
11. Feedback dado para a ação escolhida com base no resultado obtido;
12. Contagem de ciclos;
13. Contagem de vidas da criatura (reinício quando tem sucesso ou quando morre);
14. Contagem de leaflets completados (e joias capturadas);
15. Quantidade de regras aprendidas;

16. Status de execução dos ciclos.

4.9 REGISTRO DO APRENDIZADO

O resultado dos ciclos bem como as regras aprendidas e parâmetros usados para cognição da criatura são armazenados em dois arquivos:

1. Learnedrules.txt: A cada 10 ciclos os parâmetros de configuração da criatura e as regras aprendidas são gravados (sobrescritos) nesse arquivo.
2. SuccessRecord.csv: A cada vida da criatura, os resultados dessa é gravado (adicionado) nesse arquivo.



The image shows two side-by-side text editor windows. The left window, titled 'LearnedRules.txt - Bloco de notas', contains configuration parameters and learning rules. The right window, titled 'SuccessRecord.csv - Bloco de notas', contains a CSV file with learning results.

LearnedRules.txt - Bloco de notas

```
Arquivo  Editar  Formatar  Exibir  Ajuda
Forward Speed: 1  Backward Speed: 0.5  Turn Speed:0.3
BETAS>  BL: 0.5  FR: 0.5  IRL: 0.5  RER: 0.5
Generalization: 0.1  Specialization: 0.8
Learning Rate: 2
TimeBetween Cognitive Cycles: 200
Intouch Distance: 30  Fuel low level: 400
After 9923 cogcycles within 11 creature lifes, the creature completed 5 leaflets (Jewels = 21)
It has also learned the following rules:

Condition:
  Dimension Goals, Weight = 1
    (Dimension = Goals, Value = Search Food) Setting = False,
    (Dimension = Goals, Value = Search Gem) Setting = True
  Dimension VisualSensor, Weight = 1
    (Dimension = VisualSensor, Value = Gem Ahead) Setting = False,
    (Dimension = VisualSensor, Value = Food Ahead) Setting = False,
    (Dimension = VisualSensor, Value = Gem Up Right) Setting = True,
    (Dimension = VisualSensor, Value = Gem Up Left) Setting = False,
    (Dimension = VisualSensor, Value = Gem Up Front) Setting = False,
    (Dimension = VisualSensor, Value = Food Up Right) Setting = True,
    (Dimension = VisualSensor, Value = Food Up Left) Setting = False,
    (Dimension = VisualSensor, Value = Food Up Front) Setting = False

Action:
  ExternalActionChunk ROTATE_LEFT:
    Dimension SemanticLabel, Weight = 1
      (Dimension = SemanticLabel, Value = ROTATE_LEFT)

Condition:
  Dimension Goals, Weight = 1
    (Dimension = Goals, Value = Search Food) Setting = False,
```

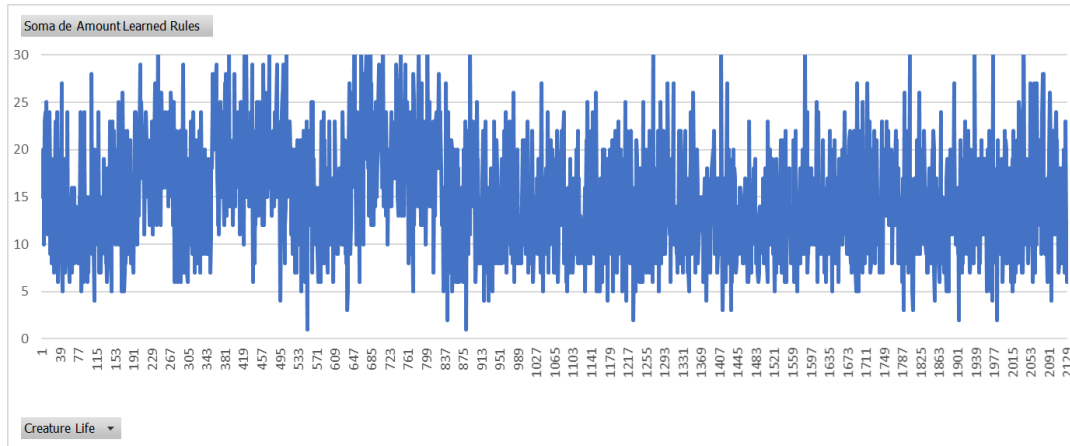
SuccessRecord.csv - Bloco de notas

```
Arquivo  Editar  Formatar  Exibir  Ajuda
Creature Life;Creature Status;Collected Jewels;
Completed Leaflets;Amount Learned Rules;
Success Rate;Time[s]

1;LOOSER;1;0;15;0;37
2;LOOSER;1;0;26;0;9
3;LOOSER;1;0;9;0;47
4;WINNER;4;1;19;0.25;16
5;LOOSER;6;1;14;0.2;37
6;WINNER;9;2;9;0.3333333333333333;22
7;WINNER;12;3;19;0.428571428571429;53
8;WINNER;15;4;16;0.5;36
9;WINNER;18;5;14;0.5555555555555556;28
10;LOOSER;19;5;10;0.5;47
```

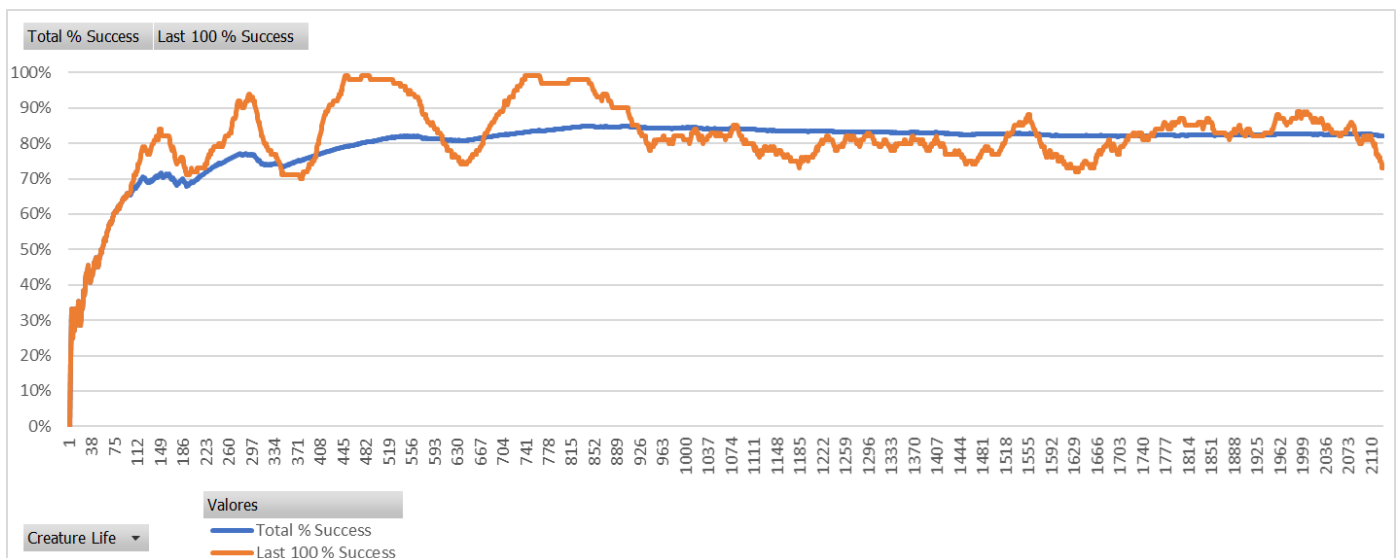
5 RESULTADOS

O controle foi executado por 1.5 milhões de ciclos, ao longo de 2129 vidas de criaturas. A quantidade de regras aprendidas flutuou ao redor de 15 ao longo desses ciclos como mostra o gráfico abaixo:



A taxa de sucesso por sua vez aumentou rapidamente, comprovando o aprendizado da criatura. A linha azul do gráfico abaixo representa a porcentagem de sucesso para o total de criaturas que viveram, enquanto a linha laranja apresenta o valor acumulado levando-se em conta apenas as últimas 100 criaturas.

Chama a atenção que embora a arquitetura tenha atingido o patamar de 99% de sucesso, durante as criaturas de vida ~450, essa taxa caiu e se estabilizando depois na faixa de 80%:



6 PRÓXIMOS PASSOS

Para melhor avaliar os resultados obtidos e possíveis de se obter com a arquitetura, outros treinamentos serão realizados com diferentes parâmetros de cognição. Também é desejável encontrar uma forma de contornar um problema encontrado onde a criatura ao encostar de lado em um objeto, ficará preso ao mesmo, recebendo sempre um feedback ruim não importando as ações que tente tomar, o que pode prejudicar consideravelmente seu aprendizado.

7 CONCLUSÃO

Através desse experimento é possível observar o potencial da arquitetura CLARION em realizar uma representação simbólica e funcional de uma cognição orgânica capaz de buscar soluções para problemas de forma não assistida.

É notável também a influência dos parâmetros de configuração da arquitetura cognitiva bem como das ações, cadeia de feedback, e inputs na capacidade e velocidade de aprendizado da criatura. Pequenas modificações nesses parâmetros e estruturas afetaram de forma desproporcional o comportamento da criatura.

Por fim, pode-se concluir que a arquitetura do CLARION é muito interessante por misturar em seu processo decisões explícitas e implícitas, e permitir que essas duas interajam entre si buscando a melhor representação possível para o conhecimento.