Action Selection – Mecanismos para a seleção de ações por agentes

Danilo Fernando Lucentini

***Abstract*— This paper aims to detail some mechanisms to model the action selection problem in intelligent autonomous agents. There will be a description of two types of mechanisms: those that include reinforcement learning and those purely reflexive.**

It will be detailed: the model of drives, network behavioral of Maes and Q-learning and R-Learning algorithms.

Besides the description of each mechanism, it will provide a brief analysis of the pros and cons of each methodology.

***Index Terms*—action selection, agents, drives, network behavioral of Maes, Q-Learning, R-Learning**

I. Introdução

Inconscientemente, todos os animais tomam centenas de decisões a cada momento. Seja uma zebra tentando fugir do seu predador ou mesmo um peixe tentando obter alimento, todos eles (independentemente do seu grau evolutivo) devem tomar ações que visam almejar um determinado objetivo.

Observando um animal na natureza, podemos pensar que esse processo de tomada de decisão pode parecer trivial; mas e se quiséssemos que um ser artificial pudesse tomar essas decisões? É basicamente nisso que consiste *action selection*.

Em linhas gerais, *action selection* é o mais básico problema relacionado a sistemas artificiais inteligentes: o que fazer a seguir? Isto é, decidir em cada momento qual a mais apropriada ação a ser executada em um conjunto de possíveis ações

Ação, no contexto de agentes autônomos, refere-se a um conjunto de entidades mutuamente exclusivas. Isto porque a demanda no atuador de um agente é tal que apenas uma ação pode ser executada em um dado instante.

É interessante ressaltar que a tentativa de implementação de algum modelo que realiza a tomada de decisão em um agente inteligente não é trivial. Várias são as considerações a serem tomadas e o próprio ambiente impõe limitações para a realização desse processo, tais como:

- Limitação de recursos, tais como hardware e software
- Inconsistência na leitura de dados dos sensores dos agentes (ruído ou má calibração)
- O mundo real é dinâmico e não estático, ou seja, a todo instante as condições estão sendo alteradas
- Objetivos dos agentes podem alterar-se ao longo do tempo. Deste modo, cabe ao agente adequar-se às novas mudanças de objetivos

Basicamente, neste trabalho elucidar-se-ão os diferentes tipos de modelos existentes para a implementação de *action selection* em agentes inteligentes. Inicialmente, será abordado um contexto mais amplo, tentando entender como o mecanismo de tomada de decisão ocorre e quais os comportamentos desejados para os modelos que visam implementá-lo. Posteriormente, serão detalhados os mecanismos mais amplamente citados na literatura, bem como seus prós e contras.

II. Características do Problema de Action Selection

Para ser possível encapsular em um agente inteligente o mecanismo de seleção de ações, é necessário primeiramente compreender como esse mecanismo ocorre na natureza.

Segundo Tyrrell [1], todo o mecanismo por trás da tomada de decisões pode ser dividido em quatro tarefas básicas:

- Percepção: sensoriamento do ambiente e interpretação dos sinais recebidos a cada momento.
- Navegação: ter ciência da própria localização da entidade e de importantes referências no meio que o rodeia.
- Seleção de ação: utilizando a percepção e os dados fornecidos pela navegação é escolhida a ação mais adequada para aquele instante de tempo
- Controle Motor: Movimentação do corpo visando realizar a ação escolhida

As tarefas anteriormente citadas norteiam o mecanismo básico para quaisquer seres vivos existentes, independentemente do seu grau de evolução. Facilmente, é possível notar que essas tarefas podem ser distribuídas a agentes inteligentes onde cada parte do agente seria designada para uma determinada tarefa. Os sensores dos agentes ficariam responsáveis pela percepção das alterações no meio que o rodeia. A tarefa de navegação pode ser realizada utilizando certos estados internos do agente, onde é possível registrar permanentemente informações relevantes de sua posição. A

seleção de ação será realizada através de algum algoritmo de *action selection* (que serão explicitados nas próximas seções). E, finalmente, o controle motor é realizado através dos atuadores dos agentes.

Antes de definir algum mecanismo para o controle de seleção de ação; é necessário, primeiramente, tomar conhecimento sobre quais propriedades esse mecanismo deve possuir. Segundo Decugis [2], para ser considerado idealmente satisfatório o mecanismo seletor de ação de um agente deve possuir:

- Reatividade: deve escolher rapidamente a sua ação dada uma mudança no meio
- Planejamento: ser capaz de predizer quais as conseqüências de sua ação e levar isso em consideração na sua escolha
- Gerenciamento de riscos: conseguir lidar com situações que impedem o agente de alcançar o seu objetivo
- Adaptação: é impossível modelar todos os comportamentos que um agente deve ter para todas as situações. Logo, é esperado que o agente seja robusto o suficiente para adaptar-se a cada cenário a fim de conquistar seu objetivo

Além disso, Patti Maes [3] acredita que os mecanismos de seleção de ações devem operar com incompleto ou mesmo incorreto conhecimento sobre o mundo e, além disso, algumas das propriedades desejadas para um agente podem ser contraditórias entre si, logo cabe ao agente balizar essas propriedades a fim de realizar seu objetivo.

III. Mecanismos para Action Selection

Após compreender um pouco melhor quais são as características almejadas para um mecanismo seletor de ações, procurar-se-á, nessa seção, citar e analisar os mais conhecidos modelos para realizar essa tarefa.

Podemos dividir esses mecanismos em dois grandes grupos: aqueles que utilizam aprendizagem por reforço, e os que não utilizam, ou seja, são apenas motivacionais. Basicamente, os mecanismos que utilizam aprendizagem por reforço procuram maximizar uma medida de desempenho baseado nos reforços (*feedbacks*) que recebe ao interagir com um ambiente desconhecido; o agente tem como objetivo aprender de maneira autônoma, através de um período de treinamento inicial, um conjunto de ações ótimas para cada situação. .Já os mecanismos que não levam em consideração a aprendizagem, apenas trabalham com os estímulos em um dado momento (externos e/ou internos), sem levar em consideração as ações anteriormente realizadas (motivacionais), isto é, são puramente reflexivos.

Nesse trabalho, serão analisados os seguintes mecanismos: Drives, Maes como mecanismos sem aprendizagem e Q-Learning e R-Learning como mecanismos que utilizam aprendizagem. Cada um desses mecanismos leva em consideração o conceito de ação já anteriormente detalhado, ou seja, cada mecanismo irá escolher para um determinado instante de tempo apenas uma possível ação que será transferida para os atuadores dos agentes.

A. *Drives*

A idéia de *drives* foi primeiramente proposta por Clark Hull [4]. Segundo Hull, os estímulos ambientais são capazes de provocar uma alteração no estímulo de um organismo. Essa alteração ele denominou de *drives*.

Em linhas gerais, *drives* são necessidades internas de uma determinada entidade e são considerados como a motivação de um comportamento ou até que ponto esse comportamento deve ser realizado. Por exemplo, quando um ser vivo sente fome, pode-se imaginar que isso desperta um *drive* de fome que o faz buscar alimento. Nesse caso fica evidente que o *drive* de fome impulsionou o ser vivo a buscar comida a fim de diminuir a necessidade interna do animal (nesse caso a fome).

Esse mesmo conceito pode ser aplicado a agentes autônomos inteligentes. Um agente pode conter diversos tipos de *drives* (obter alimento, obter água, limpeza, entre outros). A intensidade de um *drive*, que mensura o quanto aquela necessidade interna é importante para o agente, pode ser calculada utilizando diferentes tipos de funções que podem receber como parâmetro tanto o estímulo interno e/ou o estímulo externo. Em suma, pode-se dizer que:

Intensidade drive = f (estímulos internos, estímulos externos)

É interessante notar que a função que mapeia a intensidade do *drive* varia de acordo com o agente em questão. Logo, diferentes agentes terão, provavelmente, diferentes funções para os mapeamentos de seus respectivos *drives*, pois diferentes agentes têm diferentes necessidades e objetivos.

O mecanismo de seleção de ação baseado em *drives* é extremamente simples: o comportamento que possuir a maior intensidade de drive associado vence. Por exemplo, se para um determinado agente o *drive* de fome, após a execução da respectiva função de intensidade, tem valor 5 e o *drive* de fugir de predadores tem valor 3, o agente irá realizar alguma ação correspondente ao *drive* de fome pois ela é a que tem a maior intensidade entre todo o conjunto de *drives* do agente. A ação em si a ser realizada depende de cada agente, pois este mecanismo apenas seleciona qual o comportamento esperado para o sistema, fica a cargo do implementador selecionar qual a melhor ação para aquele determinado comportamento naquele momento. Para o exemplo anteriormente citado, pode-se imaginar que as possíveis ações do agente seriam: comer, ir em direção à comida ou explorar a região em busca de comida.

Vale ressaltar que a especificação da função de intensidade deve ser feita de maneira precisa, pois em alguns casos podem-se obter resultados indesejados ou ainda ocorrer o fenômeno de *dithering* descrito por Tyrell [1]. Basicamente, nesse

contexto, Tyrell coloca um agente com a mesma intensidade de *drives* para sede e para fome e a mesma distância de uma fonte de água e outra de comida. Se a função de intensidade dos *drives* não levar em consideração a distância da fonte de alimento ou água até o agente como um fator que contribuirá para o acréscimo da intensidade do *drive*, o agente poderá ir até a fonte de comida, alimentar-se, diminuir seu déficit de comida e, conseqüentemente, o seu drive de sede superará o seu drive de fome e este irá se movimentar em direção à fonte de água, portanto o agente gastará mais tempo movimentando-se da fonte de comida até a fonte de água do que efetivamente comendo ou bebendo. Se for incluído um fator externo na função de intensidade, que agora passa a levar em consideração a distância do agente até a fonte de obtenção de recursos, fica evidente que o comportamento do agente será outro, pois este gastará muito mais tempo comendo ou bebendo (realizando a ação de fato) do que com viagens entre um lugar e outro.

B. Maes

Este mecanismo foi elaborado por Petti Maes [3] e baseia-se na construção de uma rede comportamental constituída por um conjunto de módulos onde cada módulo é responsável por uma determinada tarefa, constituídos por um conjunto de ações e comportamentos

Basicamente, a rede possui dois tipos de parâmetros: estímulos externos e estímulos internos. Os estímulos externos são provenientes do ambiente e agem sobre a rede e, ao final de um processo de interações entre seus módulos, determina uma ação para o agente. Já os parâmetros internos são conexões entre os módulos constituintes da rede e contribuem para a escolha final da ação a ser executada.

Cada módulo da rede é constituído de 4 unidades:

- Lista de pré-condições que devem ser verdadeiras para o módulo tornar-se ativo.
- *Add List*: lista de condições que se tornarão ou permanecerão verdadeiras quando o módulo for executado.
- *Delete List:* lista de condições que se tornarão ou permanecerão falsas quando o módulo for executado
- Nível de ativação do módulo

Além disso, os módulos apresentam uma interligação entre si com 3 tipos de ligações possíveis:

- Sucessores: se um dado módulo A tem um módulo B como seu sucessor, então toda a preposição *p* que é membro da *add list* de A, também é membro da lista de pré-condições de B.
- Predecessores: Se um dado módulo B tem um módulo A como o seu predecessor, então existe um ligação de sucessão entre o módulo A para o módulo B (vide item anterior).
- Conflitantes: Se um dado módulo C tem um módulo D como conflitante, então para todo o membro *d* da *delete list* de C, também é membro da lista de pré-condições de D.

A idéia por trás dessa interconexão de módulos é que estes podem ativar ou inibir uns aos outros e assim o módulo selecionado garante que aquela será a melhor ação a ser tomada.

Em suma, o mecanismo de *action selection* para a rede comportamental de Maes tem como objetivo escolher o melhor módulo da rede para um determinado instante. E como isso é feito? Basicamente, um módulo para ser escolhido deve contemplar duas características: ter todas as suas pré-condições satisfeitas (diz-se nesse momento que o módulo está ativo) e ter o seu nível de ativação maior que o limiar de ativação estipulado para a rede naquele dado momento. Esse limiar de ativação começa com um valor definido e, a cada interação, recebe uma função de decaimento, diminuindo o seu valor para poder possibilitar que algum módulo torne-se executável.

Além disso, se um módulo X está sendo executado, este aumenta o nível de ativação de seus sucessores. Um módulo Z que não se encontra executável, porém ativo, aumenta a ativação de seus predecessores e, finalmente, um módulo Y que não está executável, mas ativo, diminui o nível de ativação dos módulos conflitantes com este.

Por fim, o algoritmo proposto por Petti Maes, segue os seguintes passos, que são executados continuamente para cada módulo:

1. De acordo com as condições do meio determina se o módulo se tornará ativo, isto é, se todas as pré-condições serão satisfeitas.
2. Calcula-se o nível de ativação dos outros módulos relacionados ao módulo ativo de acordo com as ligações de sucessores, predecessores e conflitantes entre eles.
3. Um módulo é executado caso contemple as seguintes condições:
   3.1. Está ativo
   3.2. O nível de ativação é maior que o limiar de ativação da rede
   3.3. O nível de ativação do módulo é maior que o de quaisquer outros módulos que contemplem as proposições 3.1 e 3.2

C. Q-Learning

Q-Learning é um método de aprendizagem introduzido por Watkins [5]. Uma das grandes vantagens desse mecanismo é que não existe a necessidade do agente saber um modelo do mundo a priori, ou seja, o agente não precisa saber detalhadamente como o ambiente que o rodeio irá agir (como é necessário em muitos outros mecanismos de aprendizagem por reforço).

A modelagem segundo o algoritmo Q-Learning é dada da seguinte forma: um agente pode possuir uma série de estados e, para cada estado, uma série de ações associadas àquele estado. Por exemplo, um agente pode possuir um estado denominado “objeto à frente”, em que existe um obstáculo impedindo o movimento do agente, e a este estado tem-se

várias ações associadas, como: desviar, ir em direção ao objeto, voltar entre outras.

Ao realizar cada ação, o agente pode passar de um estado para outro. Ou seja, tomando a mesma linha do exemplo anterior, se o agente estiver no estado de "objeto à frente" e for possível realizar a ação de desviar, pode ocorrer uma transição de estado: do estado "objeto à frente" para o estado "caminho livre", por exemplo.

Cada estado provê ao agente uma recompensa ou uma punição e o objetivo do agente é simples: maximizar a sua recompensa. Ele realizará isso aprendendo qual a melhor ação para cada estado.

Em termos matemáticos, pode-se pensar da seguinte maneira: existe um conjunto de estados para cada agente e um conjunto de ações associados aos possíveis estados. A meta do agente é sempre, a cada iteração, maximizar a sua recompensa, dessa forma dado um estado *s* e uma ação *a*, a meta do agente é sempre maximizar a função *Q(s,a)* que é a função que calcula a qualidade da combinação estado-ação. Logo, se o agente estiver no estado "objeto à frente", pode-se esperar que o resultado da função *Q(objeto à frente, desviar)* será maior que o da função *Q(objeto à frente, ir em direção ao objeto)*.

Como dito anteriormente, o algoritmo Q-Learning é um modelo de aprendizagem por reforço, logo se espera que o resultado da função *Q(s,a)* para um instante de tempo, dependa dos resultados obtidos para essa mesma função em tempos anteriores e é isso que Watkins propõe com a seguinte equação:

$$Q(s,a) = Q(s,a) + \alpha\left[r(s') + \gamma \max_{a \in A} Q(s',a) - Q(s,a)\right] \qquad (1)$$

A constante α, com $0 \le \alpha \le 1$, descreve a taxa de aprendizado, que determina o quanto as novas informações irão sobrescrever as antigas. Para uma taxa com valor 0, o agente nunca aprenderá nada, e uma taxa de 1 fará aprender apenas as ações mais recentes.

A constante $r(s')$ é a recompensa dada para o agente em um determinado estado s'. Pode ser positiva ou negativa

A constante γ, com $0 \le \gamma \le 1$, determina quanto o agente irá levar em consideração as futuras recompensas. Para um valor 0 o agente será oportunista e só considerará a recompensa atual. Esse valor multiplicará o maior valor da função *Q* entre todas as ações possíveis de um respectivo estado s'

E como o algoritmo deve interagir? Primeiramente, o algoritmo deve possuir uma tabela para as quais existem os valores pré-determinados para cada *Q(s,a)* – inicialmente, a função *Q(s,a)* pode apresentar um valor fixo escolhido a cargo do implementador.

Após essa inicialização, os passos do algoritmo são:

1. Observe o estado atual *s*
2. Escolha uma ação e execute (aleatoriamente ou utilizando alguma metodologia)
3. Observe o próximo estado s' e recompensa $r(s')$
4. Aplique os valores na equação (1) e atualize os valores da tabela *Q(s,a)*

Logo, ao final de certo período de treinamento, o agente estará habilitado a escolher a melhor ação para um determinado estado.

È interessante frisar, que apesar de ter-se encontrado a melhor ação associado a um estado, esta nem sempre deve ser a ação escolhida para evitar que o agente caia em máximos locais ou, até mesmo, resulte em um efeito de *dithering* já citado anteriormente. Uma abordagem interessante seria a escolha da melhor ação em 70 % dos casos e, nos demais casos, escolher aleatoriamente entre os outros casos ou utilizar algum outro procedimento.

D. R-Learning

O algoritmo *R-Learning* foi inicialmente proposto por Schwartz [6] e se assemelha muito ao já detalhado algoritmo *Q-Learning*. A sua principal diferença é que este algoritmo tem como meta maximizar a recompensa média a cada passo, utilizando um conceito denominado *avarage reward model*.

Analisando novamente a equação (1), do algoritmo *Q-Learning*, é possível verificar que este não usa nenhuma média para maximizar a recompensa e sim apenas descontos sucessivos para a recompensa no novo estado do agente, portanto espera-se que o algoritmo *R-Learning* obtenha melhores resultados.

A estruturação do modelo para o algoritmo *R-Learning* é muito parecida com a do algoritmo *Q-Learning*. Sua principal diferença deve-se à alteração no método de cálculo da função de qualidade da combinação estado-ação. Apenas a fins ilustrativos, denominar-se-á essa função de *R(s,a)*, ou seja, essa função retornará qual foi a qualidade da ação *a* executada para o estado *s*.

Assim, o método de cálculo para *R(s,a)* será dada pela equação:

$$R(s,a) = R(s,a) + \alpha\left[r(s') - \rho + \max_{a \in A} R(s',a) - R(s,a)\right] \qquad (2)$$

Observando atentamente a equação (2), nota-se que esta difere apenas por um fator ρ, denominado média dos reforços, que é subtraído do reforço do novo estado s' e pela não mais existência da constante γ.

O cálculo da média dos reforços é calculado segundo a equação:

$$\rho = \rho + \beta\left[r(s') + \max_{a \in A} R(s',a) - \max_{a \in A} R(s,a) - \rho\right] \qquad (3)$$

Onde β, com $0 \le \beta \le 1$, é uma constante que assegura ao agente o quanto ele irá levar em consideração as médias anteriores (semelhante ao γ utilizado no algoritmo *Q-Learning*). Essa constante multiplicará a recompensa do novo estado para o qual o agente foi transicionado, o máximo valor da função *R* para todas as ações disponíveis para o novo

estado, o máximo valor da função *R* para todas as ações disponíveis para o estado anterior à transição de estado e o valor referente à antiga média de recompensas. Inicialmente, o valor de ρ pode também ser uma constante a cargo do implementador.

Um ponto de muita importância no algoritmo *R-Learning*, é que o valor médio ρ só deve ser atualizado quando uma ação não aleatória for tomada, isto é, só se deve atualizar ρ quando a ação executada *a* for tal que:

$$R(s,a) = \max_{a \in A} R(s,a) \quad (4)$$

Assim sendo, o algoritmo deve interagir da seguinte maneira:

1. Observe o estado atual *s*
2. Escolha uma ação e execute (aleatoriamente ou utilizando alguma metodologia)
3. Observe o próximo estado s' e recompensa $r(s')$
4. Aplique os valores na equação (2) e atualize os valores da tabela R*(s,a)*
5. Se a condição da equação (4) for satisfeita, atualize ρ, utilizando a equação (3)

Novamente, ao final do período de treinamento, têm-se a melhor ação para um determinado estado, aquelas com maior *R(s,a)*, que visam maximizar a média das recompensas. Similarmente ao *Q-Learning*, essas melhores ações não devem ser executadas sempre que um agente estiver em seu respectivo estado. Ao invés disso, propõe-se utilizar alguma metodologia de escolha de ações como, por exemplo, a anteriormente citada escolha dos 70% e 30%.

IV. Análise

Na seção anterior, foram apresentados diversos mecanismos possíveis para a seleção de ação em agentes autônomos inteligentes. Agora, procurar-se-á analisar quais os benefícios e desvantagens de cada um deles.

A. Drives

Sem dúvida, a idéia principal do mecanismo é a mais simples de ser implementada. Como visto anteriormente, ele parte do princípio que estímulos internos e externos gerarão um valor para a intensidade do drive e o drive com o maior valor de intensidade será o comportamento executado pelo agente. Porém, apesar de sua simplicidade, deixa muitas lacunas sem resposta, como por exemplo: qual ação o agente deve escolher, pois esse mecanismo apenas apresenta o comportamento desejado e não a ação a ser realizada; portanto, para agentes de grande porte, a seleção da ação começa a tornar-se complexa.

Outro ponto seria a escolha da função de intensidade do drive. Como calculá-la? E quais fatores e parâmetros devem ser levados em consideração?

Fora visto anteriormente que uma função de intensidade de drive mal formulada pode gerar comportamentos desastrosos e, infelizmente, o mecanismo de drives não propõe nenhuma metodologia para a criação dessas funções, deixando a sua implementação a cargo de cada agente. Logo, o modelo de *drives* apenas sugere um comportamento e não propõe uma maneira clara e detalhada de como implementar o modelo de seleção de ação para agentes, ficando totalmente a cargo do implementador ter que modelar os diversos comportamentos do agente um a um.

B. Maes

Entre os mecanismos de seleção de ação disponíveis que não levam em consideração aprendizagem, indubitavelmente, o modelo proposto por Maes é um dos mais complexos, pois apresenta um comportamento que visa trazer para o mundo computacional uma série de ações comportamentais da natureza, como a ligação entre módulos e as ligações de ativações entre eles, por exemplo.

Porém, este modelo não é perfeito e apresenta algumas falhas. Tyrrell [1] estudou de maneira detalhada o modelo de rede comportamental proposto por Maes e, através de muitas implementações, obteve alguns resultados:

- *Deadlocks* em casos de vários objetivos: quando vários objetivos são colocados ao mesmo tempo, o agente pode ficar em um estado de *dithering*.
- Sincronismo de ação e percepção: A seleção da ação obedece a um loop síncrono: checar pré-condições de cada módulo, calcular níveis de ativação, escolher o módulo a ser executado e só então executar. O problema é que em aplicações de tempo real, não há meios de garantir que as proposições impostas no começo do loop continuarão verdadeiras ao final do procedimento, ou seja, pode-ser ter um problema de reatividade do agente.
- Um módulo é executado ou não, não existindo nenhum grau probabilístico para a seleção de uma ação.

Vários ajustes foram sugeridos para essa arquitetura, propostos por Decugis e Farber [7], por exemplo. Entretanto, segundo a visão de Tyrrell, esse modelo é muito mais adequado para problemas de planejamento de tipo e menos adequado para a reprodução da seleção de ação por animais.

C. Q-Learning e R-Learning

Esses mecanismos apresentam alta simplicidade de computação por interação, realizando pouco processamento e garantem a convergência para um máximo global, ou seja, se o nosso agente for treinado por um tempo infinito, este irá

convergir para a ação ideal em um determinado estado.

Um possível problema é a alta demanda por armazenamento de dados; pois, na pior das hipóteses, os algoritmos necessitaram algo da ordem de *s* x *a* de memória para armazenar o valor da combinação estado-ação. Porém, isso é contornável com estruturas de dados que mitigam a quantidade de dados que necessita ser armazenada.

Ambos os algoritmos partem do preceito que o mundo que os rodeia está modelado segundo um processo de decisão de Markov, ou seja, o agente está em um estado *s* e ao ser executada uma ação *a* pode ir para um estado *s'*. Em muitas situações é complexo modelar o agente em uma série de estados finitos e determinados, impossibilitando assim a utilização desses algoritmos.

Finalmente, mesmo supondo que o mundo possa ser modelado por um processo de decisão de Markov, ambos os algoritmos podem apresentar resultados inesperados caso as recompensas dadas a cada estado não sejam atribuídas de maneira correta. Assim, dependendo dos valores atribuídos, o agente pode conseguir maximizar suas recompensas de uma maneira inesperada e, em alguns casos, não desejada.

V. Conclusão

Nesse trabalho foram abordados diversos mecanismos para realizar o procedimento de *action selection* em um universo de agentes autônomos inteligentes.

Fora visto que determinar o processo de seleção de ação de um ser vivo é algo extremamente complexo e que por mais simples que uma decisão possa parecer, vários aspectos devem ser levados em cosideração.

Além disso, observando os mecanismos que esse trabalho se propôs a analisar, foi possível verificar que todos eles apresentam algum tipo de deficiência que geram dificuldades para a sua implementação na prática, ou podem gerar resultados não desejados em determinadas situações.

Entretanto, em muitos outros casos, esses algoritmos se mostraram eficientes ao contexto em que foram empregados. Logo, é de suma importância saber exatamente onde cada um dos mecanismos é aplicável e quais os impactos e conseqüências inerentes à sua aplicação.

Por fim, o problema de *action selection* está longe de ter chegado a uma solução final, com muitas lacunas pendentes para serem respondidas. Cabe a nós compreender melhor o mundo que nos rodeia e incorporar o processo de tomada de decisão dos seres vivos para o contexto de agentes inteligentes.

VI. Referências

[1] Tyrrell, T. (1993) "Computational Mechanisms for Action Selection" University of Edinburgh, Ph.D. thesis

[2] Decugis and Ferber, 1998 Decugis, V. and Ferber, J. (1998). Action selection in an autonmous agent with a hierarchical distributed reactive planning architecture. In Sycara, K. and Wooldridge, M. (eds.) Proceedings of the 2nd Int. Conference on Autonomous Agents, pages 354-361, New York, ACM Press.

[3] Maes, P. How to Learn to Do the Right Thing Submitted to the European ALIFE Coifference 1991, 1991d.

[4] Hull, C. L. (1950). Behavior postulates and corollaries— 1949. Psychological Review, 57, 173–180.

[5] Watkins, C.J.C.H. (1989) "Learning from Delayed Rewards", King's College, University of Cambridge, Ph.D. thesis

[6] A. Schwartz. A reinforcement Learning Method for Maximizing Undiscounted Rewards. In Machine Learning: Proceedings of the Tenth International Conference, San Mateo, CA, 1993. Morgan Kaufmann.

[7] Decugis, V. and Ferber, J. (1998). An extension of Maes' action selection mechanism for animats. In: From Animals to Animats 5: Proceedings of the 5th International Conference on Simulation of Adaptive Behavior (R. Pfeiffer, ed.), pp. 153-158. MIT Press.

[8] Singleton, D. (2002). An Evolvable Approach to the Maes Action Selection Mechanism

[9] Abdul Manan Ahmad, Goh Kia Eng and Tan Sang Sang. Application of action selection mechanisms in agent-based simulation