Proceedings of the International Joint Conference IBERAMIA/SBIA/SBRN 2006 - 5th Best MSc dissertation/PhD thesis contest (CTDIA'2006), Ribeirão Preto, Brazil, October 23–28, 2006. CD-ROM.
ISBN 85-87837-11-7

Uso de Heurísticas para a Aceleração do Aprendizado por Reforço

Reinaldo A. C. Bianchi[1,2] and Anna H. R. Costa[1] (orientadora)

[1] Laboratório de Técnicas Inteligentes
Escola Politécnica da Universidade de São Paulo
Av. Prof. Luciano Gualberto, trav. 3, 158. 05508-900 – São Paulo – SP, Brazil.
[2] Centro Universitário da FEI
Av. Humberto A. C. Branco, 3972. 09850-901 – São Bernardo do Campo – SP, Brazil.
rbianchi@fei.edu.br, anna.reali@poli.usp.br

Resumo Esta tese propõe uma nova classe de algoritmos que permite o uso de heurísticas para aceleração do aprendizado por reforço (AR). Esta classe de algoritmos, denominada "Aprendizado Acelerado por Heurísticas", é modelada por Processos Markovianos de Decisão, introduzindo uma função heurística $\mathcal{H}$ para influenciar um agente aprendiz na escolha de suas ações exploratórias. A heurística é usada somente para a escolha da ação a ser tomada, não modificando o funcionamento do algoritmo de AR e preservando muitas de suas propriedades. Para validar este trabalho são propostos cinco algoritmos, testados experimentalmente em diversos domínios. Os resultados experimentais permitem concluir que mesmo uma heurística muito simples resulta em uma melhora significativo do desempenho do algoritmo de AR utilizado.
Palavras Chave: Aprendizado por Reforço, Heurísticas, Robótica Móvel Inteligente.
Nível e data de conclusão: Tese de Doutorado em Engenharia Elétrica apresentada na Escola Politécnica da USP em 5 de abril de 2004.
Disponível em: http://www.teses.usp.br/teses/disponiveis/3/3141/tde-28062005-191041/

1 Introdução

Aprendizado por Reforço (AR) é uma técnica muito atraente para solucionar uma variedade de problemas de controle e planejamento quando não existem modelos disponíveis *a priori*, já que seus algoritmos têm a convergência para uma situação de equilíbrio garantida [9], além de permitirem o aprendizado de estratégias de controle adequadas.

No AR o aprendizado se dá por meio da interação direta do agente com o ambiente. Infelizmente, a convergência dos algoritmos de AR só pode ser atingida após uma extensiva exploração do espaço de estados-ações, que é geralmente demorada, inviabilizando sua utilização em grande parte dos problemas reais.

Entretanto, o uso de funções heurísticas para guiar a exploração do espaço de estados-ações pode conduzir mais rapidamente o algoritmo de AR para uma

região adequada do espaço de soluções do problema, permitindo que o agente possa atuar precocemente de manaira adequada. Este trabalho investiga como melhorar a seleção das ações por meio do uso de heurísticas em algoritmos de AR. Para validar este trabalho são propostos cinco algoritmos, testados experimentalmente em diversos domínios, como o de navegação robótica e o futebol de robôs. Os resultados experimentais permitem concluir que mesmo uma heurística muito simples resulta em um aumento significativo do desempenho do algoritmo de AR utilizado.

Este resumo descreve algumas das principais contribuições da Tese de Doutorado defendida pelo autor [2], e está organizado da seguinte maneira: a seção 2 apresenta uma visão geral do AR, mostrando um dos principais algoritmos, o *Q–Learning*. A seção 3 apresenta a formulação do "Aprendizado Acelerado por Heurísticas", proposta principal da tese. A seção 4 mostra como o aprendizado pode ser acelerado por meio do uso de heurísticas para selecionar as ações que serão realizadas durante o processo de aprendizado em uma formulação modificada do algoritmo Q-Learning. Na seção 5 são apresentados alguns dos resultados obtidos para o domínio dos robôs móveis autônomos e a seção 6 resume as principais contribuições da tese. Finalmente, a seção 7 encerra este texto com a conclusão e propostas de trabalhos futuros.

2 Aprendizado por Reforço e o algoritmo *Q–Learning*

No AR, um agente aprendiz interage com o ambiente em intervalos de tempos discretos em um ciclo de percepção-ação. A maneira mais tradicional para formalizar o AR é utilizando o conceito de Processo Markoviano de Decisão (*Markov Decision Process* – MDP), que pode ser definido formalmente [1] pela quádrupla $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$, onde: $\mathcal{S}$ é um conjunto finito de estados do ambiente; $\mathcal{A}$ é um conjunto finito de ações que o agente pode realizar; $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \Pi(\mathcal{S})$ é a função de transição de estado, onde $\Pi(\mathcal{S})$ mapeia cada par (estado, ação) em uma distribuições de probabilidades sobre o conjunto de estados $\mathcal{S}$; $T(s_t, a_t, s_{t+1})$ define a probabilidade de realizar a transição do estado s_t para o estado s_{t+1} quando se executa a ação a_t; $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \Re$ é a função de reforço.

Assim, no ciclo percepção-ação, o agente aprendiz observa, a cada passo de iteração, o estado corrente s_t do ambiente e escolhe a ação a_t para executar. Ao executar esta ação a_t – que altera o estado do ambiente – o agente recebe um sinal escalar de reforço $r_{s,a}$ (penalização ou recompensa), que indica quão desejável foi a execução de a_t em s_t. Resolver um MDP consiste em computar a política $\pi : \mathcal{S} \rightarrow \mathcal{A}$ que maximiza (ou minimiza) alguma função, geralmente a recompensa esperada (ou o custo esperado), ao longo do tempo.

Tido como o mais popular algoritmo de AR, o algoritmo *Q–Learning* foi proposto como uma maneira de aprender iterativamente a política ótima π^* quando o modelo do sistema não é conhecido [10]. O algoritmo *Q–Learning* propõe que o agente aprenda uma função Q de recompensa acumulada esperada com desconto, conhecida como função valor–ação. Ele aproxima iterativamente $\hat{Q}$ – a estimativa de $Q^*(s, a)$ no instante t – utilizando a seguinte regra de

aprendizado:

$$\hat{Q}_{t+1}(s_t, a_t) \leftarrow \hat{Q}_t(s_t, a_t) + \alpha \left[r(s_t, a_t) + \gamma \max_{a_{t+1}} \hat{Q}_t(s_{t+1}, a_{t+1}) - \hat{Q}_t(s_t, a_t) \right], \tag{1}$$

onde: s_t é o estado atual; a_t é a ação realizada em s_t; $r(s_t, a_t)$ é o reforço recebido após realizar a_t em s_t; s_{t+1} é o novo estado; γ é o fator de desconto ($0 \leq \gamma < 1$); α é a taxa de aprendizagem ($0 < \alpha < 1$).

Uma propriedade importante deste algoritmo é que as ações usadas durante o processo iterativo de aproximação da função Q podem ser escolhidas usando qualquer estratégia de exploração (ou explotação). Uma estratégia para a escolha das ações bastante utilizada em implementações do *Q–Learning* é a exploração aleatória $\epsilon - Greedy$, na qual o agente executa a ação com o maior valor de Q com probabilidade $1 - \epsilon$ e escolhe uma ação aleatória com probabilidade ϵ. Neste caso, a transição de estados é dada pela seguinte regra:

$$\pi(s_t) = \begin{cases} a_{random} & \text{se } q \leq \epsilon, \\ \arg\max_{a_t} \hat{Q}_t(s_t, a_t) & \text{caso contrário,} \end{cases} \tag{2}$$

onde q é um valor escolhido de maneira aleatória com distribuição de probabilidade uniforme sobre [0,1] e ϵ ($0 \leq \epsilon \leq 1$) é o parâmetro que define a taxa de exploração: quanto menor o valor de ϵ, menor a probabilidade de se fazer uma escolha aleatória e a_{random} é uma ação aleatória selecionada entre as ações possíveis de serem executadas no estado s_t.

3 Uso de heurísticas para aceleração do AR

O principal problema abordado neste trabalho é o da aceleração do Aprendizado por Reforço, mantendo, entretanto, propriedades interessantes e as vantagens dos algoritmos de AR, entre elas, a convergência para uma política estacionária e o aprendizado autônomo não supervisionado, ao mesmo tempo que minimiza sua principal desvantagem, que é o tempo necessário para o aprendizado.

A hipótese central deste trabalho é que existe uma classe de algoritmos de AR que permite o uso de heurísticas para abordar o problema da aceleração do aprendizado. Esta classe de algoritmos é aqui denominada "Aprendizado Acelerado por Heurísticas" (*Heuristically Accelerated Learning* – HAL).

Uma heurística pode ser definida como uma técnica que melhora, no caso médio, a eficiência na solução de um problema. Segundo Russell e Norvig, "funções heurísticas são a forma mais comum de se aplicar o conhecimento adicional do problema a um algoritmo de busca" [8], sendo, dessa forma, uma maneira de generalização do conhecimento que se tem acerca de um domínio.

De maneira mais formal, um algoritmo da classe HAL pode ser definido como um modo de solucionar um problema modelável por um MDP que utiliza explicitamente a função heurística $\mathcal{H} : \mathcal{S} \times \mathcal{A} \rightarrow \Re$ para conduzir o agente na escolha de suas ações, durante o aprendizado. $H_t(s_t, a_t)$ define a heurística que indica a importância de se executar a ação a_t estando no estado s_t.

A função heurística está fortemente vinculada à política: a heurística indica que uma ação deve ser tomada em detrimento de outra. Dessa forma, pode-se dizer que a função heurística define uma "Política Heurística", isto é, uma política exploratória que visa acelerar o aprendizado.

A utilização da função heurística feita pelos HALs explora uma característica importante do AR: a livre escolha das ações de treino. Desta forma, pode-se dizer que os algoritmos de AR são um subconjunto dos algoritmos HAL, onde a influência da heurística é sempre nula. Uma heurística adequada acelera o aprendizado e, se a heurística não for adequada, o resultado não impede o aprendizado de convergir para um valor estacionário ótimo (no caso dos sistemas deterministas). Dessa maneira, pode-se construir algoritmos da classe HAL a partir de qualquer algoritmo de AR. Por não modificar o funcionamento do algoritmo de AR, esta proposta permite que muitas das conclusões obtidas para algoritmos de AR continuem válidas para os HALs.

Diversas abordagens para a aceleração do AR foram propostas nos últimos anos, sendo que a maioria ou realiza um melhor aproveitamento das experiências, por meio de generalizações – temporais, espaciais ou das ações – ou utiliza abstrações – temporais ou estruturais. Um algoritmo bem conhecido que utiliza generalização temporal é o $Q(\lambda)$ [10], que estende o *Q–Learning* adicionando a este uma propagação das atualizações utilizando o conceito de elegibilidade de um par (estado,ação). A elegibilidade reflete a quantidade de vezes que um par foi visitado no passado recente. O uso de elegibilidade permite que um reforço recebido seja usado para atualizar todos os estados recentemente visitados, fazendo com que os estados mais próximos das recompensas sejam mais influenciados por elas.

A generalização espacial envolve a distribuição dos resultados de uma experiência para vários estados, segundo alguma medida de similaridade do espaço de estados. Um algoritmo que realiza este espalhamento é o QS [7]. Nele, o algoritmo *Q–Learning* é combinado com o espalhamento espacial na função valoração. Assim, ao receber um reforço, outros pares valor-ação que não estavam envolvidos na experiência também são atualizados, utilizando o conhecimento que se tem acerca das similaridades no domínio.

A idéia de utilizar heurísticas em conjunto com um algoritmo de aprendizado já foi abordada por outros autores, como na abordagem de Otimização por Colônia de Formigas [5], proposta para resolver o Problema do Caixeiro Viajante, no qual diversas formigas viajam entre cidades e o trajeto mais curto é reforçado. Porém, as possibilidades do uso de heurísticas ainda não foram devidamente exploradas. A seguir é feita uma análise mais aprofundada de cada elemento dos algoritmos HAL.

3.1 A função heurística $\mathcal{H}$

A função heurística explora o conhecimento sobre uma política apropriada para acelerar o aprendizado, conhecimento este derivado diretamente do domínio ou construído a partir de indícios existentes no próprio processo de aprendizagem.

A função heurística pode atuar na política de exploração aleatória $\epsilon-Greedy$, na qual, além da estimativa das funções de probabilidade de transição $\mathcal{T}$ e da recompensa $\mathcal{R}$, a função $\mathcal{H}$ também é considerada. Assim, a regra de transição de estados aqui proposta é dada por:

$$\pi(s_t) = \begin{cases} a_{random} & \text{se } q \leq \epsilon, \\ \arg\max_{a_t} \left[\mathsf{F}_t(s_t, a_t) \bowtie \xi H_t(s_t, a_t)^{\beta} \right] & \text{caso contrário,} \end{cases} \tag{3}$$

onde:

- $\mathcal{F} : \mathcal{S} \times \mathcal{A} \rightarrow \Re$ é uma estimativa da função valor que descreve a recompensa esperada acumulada. Por exemplo, se $\hat{\mathsf{F}}_t(s_t, a_t) \equiv \hat{Q}_t(s_t, a_t)$ tem-se um algoritmo similar ao *Q–Learning*;
- $\mathcal{H} : \mathcal{S} \times \mathcal{A} \rightarrow \Re$ é a função heurística, que influencia a escolha da ação. $H_t(s_t, a_t)$ define a importância de se executar a ação a_t estando no estado s_t. O índice t na função heurística indica que ela pode ser diferente em dois instantes distintos;
- $\bowtie$ é uma função matemática, que deve operar sobre números reais e produzir um valor pertencente a um conjunto ordenado (que suporte a operação de maximização);
- ξ e β são variáveis reais usadas para ponderar a influência da função heurística;
- q é um valor escolhido de maneira aleatória com distribuição de probabilidade uniforme sobre [0,1] e ϵ ($0 \leq \epsilon \leq 1$) é o parâmetro que define a taxa de exploração: quanto menor o valor de ϵ, menor a probabilidade de se fazer uma escolha aleatória;
- a_{random} é uma ação selecionada de maneira aleatória entre as ações executáveis no estado s_t.

A principal conseqüência desta formulação é que as provas de convergência existentes para os algoritmos de AR continuam válidas nesta abordagem. O teorema a seguir é um dos três teoremas apresentados por Bianchi [2] que confirmam esta afirmação e limitam o erro máximo causado pelo uso de uma heurística.

Teorema: *Se a técnica de Aprendizado por Reforço na qual um HAL é baseado corresponde a uma forma generalizada do algoritmo* Q–Learning *[9], então o valor* $\hat{\mathsf{F}}_t$ *converge para o* F^**, maximizando a recompensa acumulada esperada com desconto, com probabilidade unitária para todos os estados* $s \in \mathcal{S}$*, desde que a condição de visitação infinita de cada par (estado,ação) seja mantida.*

Esboço da prova: Nos algoritmos HAL, a atualização da aproximação da função valor ao valor estacionário não depende explicitamente do valor da heurística. As condições necessárias para a convergência dos algoritmos *Q–Learning* generalizados que podem ser afetadas com o uso da heurística nos algoritmos HAL são as que dependem da escolha da ação. Das condições apresentadas por Szepesvari e Littman [9], a única que depende da ação escolhida é a necessidade de visitação infinita a cada par (estado, ação). Como a equação 3 propõe uma estratégia de exploração ϵ*–greedy* que leva em conta a aproximação da função valor $\mathsf{F}(s_t, a_t)$

já influenciada pela heurística, a possibilidade de visitação infinita a cada par (estado, ação) é garantida e o algoritmo converge. □

A condição de visitação infinita de cada par (estado, ação) pode ser aceita na prática – da mesma maneira que ela é aceita para os algoritmos de AR em geral – por meio do uso de diversas outras estratégias de visitação: utilizando uma estratégia de exploração Boltzmann [6] ao invés da exploração ϵ–*greedy*; intercalando passos onde se utiliza a heurística com passos de exploração; iniciando cada novo episódio a partir dos estados menos visitados; ou utilizando a heurística durante um período determinado de tempo, menor que o tempo total de exploração, permitindo que o agente ainda visite estados não apontados pela heurística.

3.2 Definição da Função Heurística $\mathcal{H}$

A função heurística $\mathcal{H}$ pode ser definida de maneira *ad hoc* ou pode ser extraída automaticamente, com base em casos similares ou durante o aprendizado. Para a extração automática das heurísticas, uma classe de métodos é proposta. Estes métodos funcionam geralmente em dois estágios. O primeiro, que retira da estimativa da função valor F informações sobre a estrutura do domínio e o segundo, que encontra a heurística para a política – em tempo de execução ou a partir de uma base de casos – usando as informações extraídas de F. Estes estágios foram aqui denominados de "Extração de Estrutura"e "Construção da Heurística", respectivamente.

Foram estudadas diversas maneiras possíveis de extrair a estrutura de um domínio. Uma delas é o método "Estrutura a partir de Exploração", que constrói iterativamente uma estimativa da função de probabilidade de transição de estados, anotando o resultado de todas as ações realizadas pelo agente.

Um novo método automático para compor a heurística em tempo de execução, a partir da estrutura do domínio extraída, é proposto neste trabalho e é chamado "Retropropagação de Heurísticas". Ele propaga, a partir de um estado final, as políticas corretas que levam àquele estado. Por exemplo, ao chegar no estado terminal, define-se a heurística como composta pelas ações que levam de estados imediatamente anteriores a este estado terminal. Em seguida, propaga-se esta heurística para os antecessores dos estados que já possuem uma heurística definida. A combinação do método de extração de Estrutura a partir de Exploração com a Retropropagação de Heurísticas gera o método "Heurística a partir de Exploração".

4 O algoritmo *Q–Learning* Acelerado por Heurísticas

Por ser o mais popular algoritmo de AR e possuir uma grande quantidade de dados na literatura para a realização de uma avaliação comparativa, exemplifica-se um algoritmo da classe HAL por uma extensão do algoritmo *Q–Learning*. Este novo algoritmo é denominado "*Q–Learning* Acelerado por Heurísticas" – HAQL.

Para sua implementação, é necessário definir a regra de transição de estados e o método a ser usado para atualizar a heurística. A regra de transição de estados, definida na Equação 3 é, no caso do HAQL, dada por:

$$\pi(s_t) = \begin{cases} a_{random} & \text{se } q \leq \epsilon, \\ \arg\max_{a_t} \left[\hat{Q}(s_t, a_t) + \xi H_t(s_t, a_t)\right] & \text{caso contrário.} \end{cases} \quad (4)$$

Vale notar que as únicas modificações em relação ao algoritmo *Q–Learning* se referem ao uso da função heurística para a escolha da ação a ser executada e a existência de um passo de atualização da função $H_t(s_t, a_t)$. A convergência deste algoritmo é demonstrada em [2] .

5 Experimentos com o algoritmo HAQL

O objetivo desta seção é verificar o desempenho do HAQL ao ser aplicado na simulação de um robô real imerso em um ambiente não determinista e sujeito a erros de posicionamento. Para tanto, foi utilizada a plataforma Saphira controlando um robô móvel Pioneer 2DX, fabricado pela ActivMedia Robotics.

Para realizar o aprendizado, o ambiente foi particionado em células, cada uma aproximadamente do tamanho do robô. O ambiente utilizado – um quadrado de 10×10 metros – foi discretizado em uma grade de 20 x 20 células. A orientação do robô foi particionada em 16 valores discretos, que englobam 20 graus cada. Assim, a variável de estado do robô (x, y, θ), usada pelo aprendizado, é uma representação grosseira da postura real do robô. Foram definidas quatro ações possíveis de serem executadas pelo robô: mover para frente ou para trás, por uma distância correspondente ao tamanho do robô, e girar, no próprio eixo, 20 graus no sentido horário ou anti-horário. Ai início de um episódio, o robô se encontra no canto inferior esquerdo e o alvo que ele deve alcançar se encontra no canto superior direito.

Neste domínio, foi realizada a comparação entre o *Q–Learning* e o HAQL utilizando o método "Heurística a partir da Exploração" para acelerar o aprendizado a partir do 5.^{o} episódio. Toda vez que passa por uma célula, o robô anota a passagem, computando as visitas às células. Ao final do quinto episódio, esta computação é limiarizada, resultando em um esboço do mapa do ambiente, usado para criar a heurística por meio da "Retropropagação de Heurísticas". (ver figura 1). O mapa não corresponde exatamente ao ambiente de teste, resultando em uma heurística que não indica exatamente a política ótima.

Os resultados, que mostram a média de 30 treinamentos, podem ser vistos na tabela 1. Nela, é apresentado o número de passos (média e desvio padrão) até se atingir o alvo para o *Q–Learning* (na 2^a coluna) e o HAQL (3^a coluna). Foi também utilizado o teste T de Student para verificar se é válida a hipótese de que o uso do algoritmo HAQL acelera o aprendizado. O resultado do teste T de Student para este experimento mostra que, a partir do 6.^{o} episódio, os resultados são significativamente diferentes, com nível de confiança próximo a 0,01% (tabela 1, 4^a e 5^a colunas).

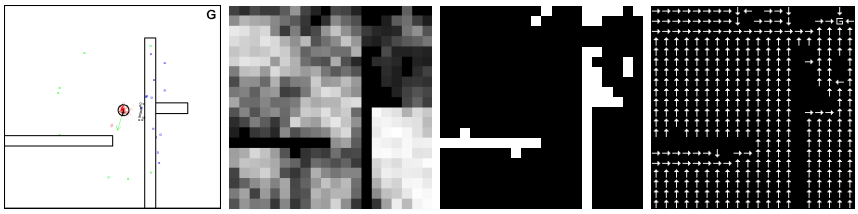

Figura 1. Da esquerda para a direita: Ambiente de teste (onde o círculo indica o robô e 'G' o alvo), número de visitas (branco indica um número maior de visitas), o esboço do mapa criado utilizando o método "Estrutura a partir de Exploração" para o ambiente Saphira, e a heurística criada.

Episódio	Q–Learning (passos)	HAQL (passos)	Módulo de T	Nível de Confiança
6	7902 ± 8685	65 ± 51	4.942	0.01%
7	10108 ± 10398	55 ± 43	5.295	0.01%
8	8184 ± 12190	72 ± 61	3.644	0.02%
9	8941 ± 8367	75 ± 71	5.803	0.01%
10	8747 ± 9484	63 ± 55	5.015	0.01%

Tabela 1. Resultado da aceleração a partir do sexto episódio usando o algoritmo HAQL no ambiente de simulação Saphira.

Finalmente, a figura 2 mostra os caminhos percorridos pelo robô utilizando o *Q–Learning* e o HAQL. Nas duas figuras, geradas com o resultado do sexto episódio de treinamento de ambos os algoritmos (portanto, com a heurística atuando no HAQL), o robô inicia no canto esquerdo inferior e deve atingir a meta localizada no canto direito superior. Pode-se notar que, utilizando o *Q–Learning*, o robô caminha aleatoriamente até encontrar a meta, executando 12081 passos para atingi-la, enquanto que, ao usar o HAQL, ele se dirige quase certeiramente para o alvo, executando 86 passos. Neste exemplo pode-se ver claramente que a ação da heurística é a de limitar a exploração no espaço de busca, direcionando a navegação do robô no ambiente, mas ainda permitindo pequenas explorações. Constata-se, neste experimento, a melhora do desempenho do algoritmo de AR com o uso de heurísticas.

6 Principais contribuições

As principais contribuições deste trabalho [2, 3, 4] são:

- A formalização da classe de algoritmos de Aprendizado Acelerado por Heurísticas com a introdução de uma função heurística $\mathcal{H}$ que influencia a escolha das ações e é atualizada durante o processo de aprendizado, preservando, no entanto, muitas das propriedades dos algoritmos de AR.
- A verificação do fato que muitas das propriedades dos algoritmos de AR também são válidas para os HALs. Foi mostrado que as provas de convergência existentes para os algoritmos de AR correspondentes a uma forma

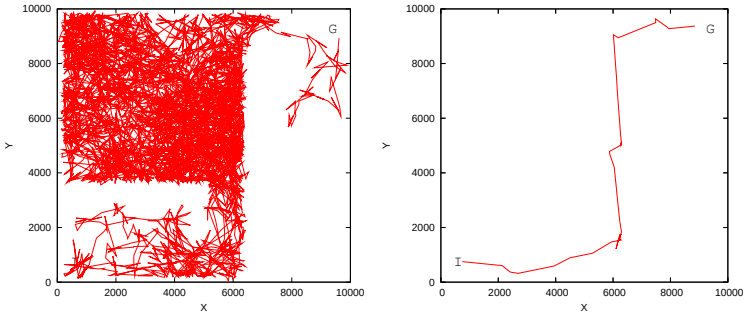

Figura 2. Exemplos de caminhos percorridos pelo robô utilizando o *Q–Learning* (esquerda) e o HAQL (direita) no ambiente de simulação Saphira 8.0, no 6^o episódio.

generalizada do algoritmo *Q–Learning* continuam válidas nesta abordagem e foi calculado o erro máximo causado pelo uso de uma heurística.
- A proposta de métodos automáticos de extração da função heurística $\mathcal{H}$, a partir do domínio do problema ou do próprio processo de aprendizagem. De maneira geral, os métodos do segundo tipo funcionam em dois estágios: o primeiro retira da estimativa da função valor informações sobre a estrutura do domínio e o segundo encontra uma heurística para a política usando as informações extraídas; estes estágios foram denominados de Extração de Estrutura e Construção da Heurística, respectivamente.
- A comparação do uso da função heurística $\mathcal{H}$ pelos HALs com o uso de heurísticas em algoritmos de busca informada.
- A verificação de que as informações existentes no domínio ou em estágios iniciais do aprendizado permitem definir a função heurística $\mathcal{H}$. Entre os indícios existentes no processo de aprendizagem, os mais relevantes são a função valor em um determinado instante, a política do sistema em um determinado instante e o caminho pelo espaço de estados que o agente pode explorar.

Foram propostos e estudados cinco algoritmos de AR acelerados por heurísticas, que estendem algoritmos de AR bem conhecidos: o algoritmo *Heuristically Accelerated Q–Learning* – HAQL, apresentado neste resumo; o *Heuristically Accelerated* SARSA(λ); o *Heuristically Accelerated* TD(λ); o *Heuristically Accelerated Distributed Q–Learning* e finalmente, o *Heuristically Accelerated Minimax–Q*.

Todos os algoritmos propostos apresentaram um desempenho significativamente melhor que os algoritmos originais, sendo esta conclusão baseada nos resultados do teste T de Student. Foram realizados testes em diversos domínios bem conhecidos na literatura de AR, permitindo verificar a generalidade de atuação dos HALs. Além dos agentes robóticos inteligentes, outros domínios, cujos resultados também se aplicam à robótica, foram estudados: o carro na montanha, o pêndulo invertido, o problema do caixeiro viajante e o futebol de robôs simulado.

7 Conclusão e trabalhos futuros

Este trabalho propôs uma classe de algoritmos que permite o uso de heurísticas para aceleração do AR. Os resultados obtidos indicam que esta classe de algoritmos, denominada "Aprendizado Acelerado por Heurísticas" (*Heuristically Accelerated Learning* – HAL), permite a extensão dos algoritmos existentes de AR, tornando-os mais eficientes.

A principal vantagem do uso da heurística é que esta limita o espaço de busca que o agente aprendiz explora nos estágios iniciais do aprendizado, conduzindo-o a um espaço de soluções mais adequado. Neste sentido, a heurística funciona, no aprendizado, de maneira similar aos algoritmos de busca informada: indica o melhor caminho a seguir, reduzindo a busca. Os HALs apresentam ainda a vantagem de, por utilizarem a heurística em conjunto com o aprendizado, poderem aprender a superar heurísticas ruins e encontrar o melhor caminho. No caso do uso de heurísticas inadequadas, pode ocorrer um atraso no aprendizado ou mesmo uma aceleração, após uma piora pontual no desempenho do agente.

Duas questões que foram estudadas superficialmente e ainda exigem maior atenção são as duas primeiras tarefas que se pretende realizar como extensão deste trabalho: estudar métodos que permitam reutilizar conhecimentos aprendidos *a priori* para acelerar o aprendizado e estudar maneiras de compartilhar o conhecimento entre diversos agentes para acelerar o aprendizado.

Referências

[1] D. P. Bertsekas. *Dynamic Programming: Deterministic and Stochastic Models*. Prentice-Hall, Upper Saddle River, NJ, 1987.

[2] R. A. C. Bianchi. *Uso de Heurísticas para a Aceleração do Aprendizado por Reforço*. Tese de Doutorado em Engenharia Elétrica, 171p. Escola Politécnica da Universidade de São Paulo, 2004.

[3] R. A. C. Bianchi, C. H. C. Ribeiro, and A. H. R. Costa. Heuristically accelerated q-learning: a new approach to speed up reinforcement learning. *Lecture Notes in Artificial Intelligence*, 3171:245–254, 2004.

[4] R. A. C. Bianchi, C. H. C. Ribeiro, and A. H. R. Costa. Accelerating autonomous learning by using heuristic selection of actions. *Journal of Heuristics*, (no prelo) 2006.

[5] E. Bonabeau, M. Dorigo, and G. Theraulaz. Inspiration for optimization from social insect behaviour. *Nature 406 [6791]*, 2000.

[6] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

[7] C. H. C. Ribeiro. Embedding a priori knowledge in reinforcement learning. *Journal of Intelligent and Robotic Systems*, 21:51–71, 1998.

[8] S. Russell and P. Norvig. *Inteligência Artificial*. Elsevier, Rio de Janeiro, 2a. edição, 2004.

[9] C. Szepesvári and M. L. Littman. Generalized markov decision processes: Dynamic-programming and reinforcement-learning algorithms. Technical report, Brown University, 1996. CS-96-11.

[10] C. J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, University of Cambridge, 1989.