Universidade de São Paulo

Escola de Artes, Ciências e Humanidades

Gabriel Onibeni Pelussi - 7971862

Vitor Manhaes D'Amico - 8061751

Marcos Pinheiro Moura - 7971841

Relatório EP de Inteligência Artificial

Prof. Dr. Valdinei Freire da Silva

São Paulo

2016

Value iteration:

Este algorítimo itera sobre a a função valor de um ambiente uma quantidade arbitrária de vezes até que a diferença entre duas funções valor seja menor do que um valor arbitrário. A função valor pode ser dada por V* onde:

$$
\begin{aligned}
Q^*(s,a) &= \sum_{s'} P(s'|a,s)\left(r(s,a,s') + \gamma V^*(s')\right) \\
V^*(s) &= \max_a Q^*(s,a) \\
\pi^*(s) &= \arg\max_a Q^*(s,a)
\end{aligned}
$$

Fig. 1[1]

Isso quer dizer que a função Q é dada pela equação de Bellman, utilizando a probabilidade de transição de cada estado para cada ação, a função recompensa da transição de um estado para seu estado meta através de uma determinada ação, um *y* (determinado 0.9 para este EP) e a função valor anterior da iteração anterior. A função valor é dada pelos maiores valores do vetor *Q* para cada estado e a política é formada pelas ações que geraram tais valores para da função valor para cada estado.
Ficamos com o seguinte pedaço de código representando a função de Bellman:

```
for a=1:A % para cada ação
% Q(s->s')=R(s->s') + 0.9  * P(s->s') * Vanterior;
  Q(:, a) = R(:, a) + gama * T{a}    * Vanterior;
end;
% V = às maiores recompensas
% politica = as ações que levaram a tais recompensas
(índices de V)
[V, politica] = max(Q, [], 2);
```

O resto do código é apenas a comparação entre a variação que ocorreu entre cada iteração e o valor arbitrário *delta*:

```
gama = 0.9
epsilon = 0.01
delta = epsilon * (1-gama)/gama % Limite da variação
variacao = max(V - Vanterior) - min(V - Vanterior);
if variacao < delta;
     terminou = true;
end;
```

Policy iteration:

O algorítimo de policy iteration funciona de maneira muito semelhante e existem implementações modificadas e mais heurísticas do que a abordada no EP. A policy iteration também utiliza da equação de Bellman para, mas para observar modificações na política e usar a mesma para com comparação e condição de parada. A iteração da política deve fazer com que ela convirja até uma política ideal onde ela não altera entre uma iteração e outra. Sendo assim, quando tivermos:

```
política_anterior == política
>> true
```

então teremos a política ideal alcançada através da iteração de políticas.

Resultado:

Ambos os algorítimos obtiveram resultados idênticos no *ambienteTeste*, uma diferença de apenas 3 ações no *ambiente1* e o *ambiente2* obteve cerca de 60 ações diferentes para as políticas resultantes. O *policy iteration* iterou muito mais vezes à medida que os ambientes cresceram. Entre 3 e 7 vezes mais entre os diferentes ambientes.

Conclusão:

Dado o número de estados para as políticas de cada ambiente, acreditamos que a diferença entre os algorítimos não foi significante. Infelizmente não conseguimos descobrir o resultado de algum dos algorítimos realmente é melhor que o outro efetivamente em valor, mas descobrimos que *policy iteration* é mais custoso (maior número de iterações) e itera até que realmente não consiga melhorar em nada a política, tornando-o, teoricamente, capaz de resultar em uma melhor política já que ambos geram políticas através do mesmo algorítimo de Bellman e apenas o faz até não o conseguir mais, mas o *value iteration* não fica muito para trás e tem um custo relativamente baixo comparado ao outro, o que pode ter importância em ambientes com centenas de milhares ou milhões de estados sendo executados em computadores pessoais (sic).

Bibliografia:

1:Decision theory, value iteration, Disponível em
https://www.cs.ubc.ca/~kevinlb/teaching/cs322%20-%202008-9/Lectures/DT4.pdf
Acesso em 15 de maio de 2016

Artificial Inteligence, Foundation of computational agents, Disponível em
http://artint.info/html/ArtInt_227.html
http://artint.info/html/ArtInt_228.html
Acesso em 15 de maio de 2016

Markov decision process (MDP) Toolbox for matlab, Disponível em
http://www.cs.ubc.ca/~murphyk/Software/MDP/mdp.html
Acesso em 17 de maio de 2016

Stack overflow, disponivel em
http://stackoverflow.com/questions/8337417/markov-decision-process-value-iteration-how-does-it-work
Acesso em 18 de maio de 2016

Markov decision process and Bellman Equations, Todorov, Emo. Disponível em
https://homes.cs.washington.edu/~todorov/courses/amath579/MDP.pdf
Acesso em 18 de maio de 2016

Math Works, disponivel em
http://www.mathworks.com/help/matlab/
Acesso em 18 de maio de 2016