SimpleMDPLibrary

Generated by Doxygen 1.8.11

Contents

Chapter 1

Simple MMDP Library - A simple Markov Decision Process Library.

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Class Documentation

4.1 MDP Struct Reference

A Markov Decision Process.

`#include <mdp.h>`

Public Attributes

- int s
- int a
- float gamma
- float *** P
- unsigned char * t
- float * r
- float * v
- int * pi

4.1.1 Detailed Description

A Markov Decision Process.

Structure containing all parameters, infinite horizon expected rewards and optimal policy.

4.1.2 Member Data Documentation

4.1.2.1 int MDP::a

number of actions

4.1.2.2 float MDP::gamma

discount factor

4.1.2.3 float*** MDP::P

transition probabilities, P(s'|s,a)(access: P[s'][s][a])

4.1.2.4 int* MDP::pi

the optimal policy

4.1.2.5 float* MDP::r

the reward r(s)

4.1.2.6 int MDP::s

number of states

4.1.2.7 unsigned char* MDP::t

terminal states

4.1.2.8 float* MDP::v

the infinite horizon expected rewards

The documentation for this struct was generated from the following file:

- mdp.h

Chapter 5

File Documentation

5.1 mdp.c File Reference

Simple MDP library.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <float.h>
#include <math.h>
#include "mdp.h"
```

Include dependency graph for mdp.c:

5.2 mdp.h File Reference

Simple MDP library.

This graph shows which files directly or indirectly include this file:

Classes

- struct MDP

  A Markov Decision Process.

Functions

- MDP * allocate_MDP (int s, int a, float gamma)

  Allocates memory for an MDP.
- void release_MDP (MDP *mdp)

  Releases memory for an MDP.
- void print_MDP (MDP *mdp)

  Prints the MDP parameters.
- void random_MDP (MDP *mdp)

  Set random probabilities for the state transitions.

- void v_iteration_compute (MDP *mdp, float epsilon)

  Compute v(s) using value iteration.*
- float best_expected_u (MDP *mdp, int s)

  Computes the best expected utility value for the state s.
- int best_expected_action (MDP *mdp, int s)

  Computes the action for the best expected utility value for the state s.
- float expected_u (MDP *mdp, int s, int a)

  Computes the expected utility value for the state s when taking action a.
- float error (MDP *mdp, float *v)

  Computes the maximum error between the values of the MDP and the given vector v.
- void compute_optimal_policy (MDP *mdp)

  Computes the optimal policy for the given MDP.

5.2.1 Detailed Description

Simple MDP library.

Author

Stalin Muñoz Gutiérrez

Date

27 may 2018 Simple Markov Decision Process library header file

5.2.2 Function Documentation

5.2.2.1 MDP* allocate_MDP (int *s*, int *a*, float *gamma*)

Allocates memory for an MDP.

Parameters

s	the number of states
a	the number of actions
gamma	the discount factor for future rewards

5.2.2.2 int best_expected_action (MDP * *mdp*, int *s*)

Computes the action for the best expected utility value for the state s.

Parameters

mdp	the MDP
s	the state for the computation

5.2.2.3 float best_expected_u (MDP * *mdp*, int *s*)

Computes the best expected utility value for the state s.

Parameters

mdp	the MDP
s	the state for the computation

5.2.2.4 void compute_optimal_policy (MDP * *mdp*)

Computes the optimal policy for the given MDP.

Parameters

mdp	the MDP

5.2.2.5 float error (MDP * *mdp*, float * *v*)

Computes the maximum error between the values of the MDP and the given vector v.

Parameters

mdp	the MDP
v	the vector of values for the computation

5.2.2.6 float expected_u (MDP * *mdp*, int *s*, int *a*)

Computes the expected utility value for the state s when taking action a.

Parameters

mdp	the MDP
s	the state for the computation
a	the action to take

5.2.2.7 void print_MDP (MDP * *mdp*)

Prints the MDP parameters.

Parameters

mdp	the MDP to print

5.2.2.8 void random_MDP (MDP * *mdp*)

Set random probabilities for the state transitions.

Parameters

mdp	the MDP

5.2.2.9 void release_MDP (MDP * *mdp*)

Releases memory for an MDP.

Parameters

the	MDP to release

5.2.2.10 void v_iteration_compute (MDP * *mdp*, float *epsilon*)

Compute v*(s) using value iteration.

Parameters

mdp	the MDP
epsilon	the maximum allowed error for the computation of v(s)

5.3 random_example.c File Reference

Example with a random instance of a 3 state MDP.

```
#include <stdio.h>
#include "mdp.h"
```

Include dependency graph for random_example.c:

Functions

- void **random_example** ()
- int main (int argc, char **argv)

  Simple invocation of a random instance of an MDP.

5.3.1 Detailed Description

Example with a random instance of a 3 state MDP.

Author

Stalin Muñoz Gutiérrez

Date

27 may 2018

5.3.2 Function Documentation

5.3.2.1 int main (int *argc,* char ** *argv*)

Simple invocation of a random instance of an MDP.

Parameters

argc	number of command line params
argv	no params for this example

5.4 ryn_chapter_17.c File Reference

Simple MDP library.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <float.h>
#include <math.h>
#include "mdp.h"
```

Include dependency graph for ryn_chapter_17.c:

Functions

- void **ryn_chapter_17** ()
- int main (int argc, char **argv)

  Runs the Russell & Norvig chapter 17 example Russell, Stuart J., and Peter Norvig. Artificial intelligence: a modern approach. Pearson Education Limited, 2016.

5.4.1 Detailed Description

Simple MDP library.

Author

Stalin Muñoz Gutiérrez

Date

28 may 2018 Simple Markov Decision Process library implementation.

5.4.2 Function Documentation

5.4.2.1 int main (int *argc,* char ** *argv*)

Runs the Russell & Norvig chapter 17 example Russell, Stuart J., and Peter Norvig. Artificial intelligence: a modern approach. Pearson Education Limited, 2016.

Parameters

argc	number of command line params
argv	no params for this example

Index