Package 'pomdp'

May 14, 2019

Title Solver for Partially Observable Markov Decision Processes (POMDP)

Version 0.9.1-1

Date 2019-05-14

Description Provides an interface to pomdp-solve, a solver for Partially Observable Markov Decision Processes (POMDP). The package enables the user to simply define all components of a POMDP model and solve the problem using several methods. The package also contains functions to analyze and visualize the POMDP solutions (e.g., the optimal policy).

Depends R (>= 3.1.0)

License GPL (>= 3)

Suggests knitr, rmarkdown

VignetteBuilder knitr

LazyData true

Imports igraph

Copyright pomdp-solve is Copyright (C) Anthony R. Cassandra; LASPack is Copyright (C) Tomas Skalicky; lp-solve is Copyright (C) Michel Berkelaar, Kjell Eikland, Peter Notebaert; all other code is Copyright (C) Hossein Kamalzadeh and Michael Hahsler.

NeedsCompilation yes

Author Hossein Kamalzadeh [aut, cph, cre], Michael Hahsler [aut, cph], Anthony R. Cassandra [ctb, cph]

Maintainer Hossein Kamalzadeh <hkamalzadeh@smu.edu>

Repository CRAN

Date/Publication 2019-05-14 21:40:08 UTC

R topics documented:

`model` *Extract the User-defined Model Components from a Solved POMDP*

Description

The function returns the POMDP model components of a solved POMDP.

Usage

```
model(x)
```

Arguments

`x`	object of class POMDP returned by `solve_POMDP`.

Value

An object of class "POMDP_model", i.e., a list of all model components.

See Also

`solve_POMDP`

Examples

```
data("TigerProblem")
tiger_solved <- solve_POMDP(model = TigerProblem)
tiger_solved

model(tiger_solved)
```

plot *Visualize a POMDP Policy Graph*

Description

The function plots the POMDP policy graph in an object of class POMDP. It uses `plot` in **igraph** with appropriate plotting options to show information like the belief proportions (if available).

Usage

```
## S3 method for class 'POMDP'
plot(x, y = NULL, vertex.size = 40, edge.arrow.size =.5,
  vertex.frame.color = "grey", ...)
```

Arguments

`x`	object of class POMDP.
`y`	ignored.
`vertex.size, edge.arrow.size, vertex.frame.color, ...`	plotting options passed on to `plot.igraph` in **igraph** (see `plot.common` for available options).

See Also

`solve_POMDP`, `plot.igraph`, `igraph_options`, `plot.common`

Examples

```
data("TigerProblem")
tiger_solved <- solve_POMDP(model = TigerProblem)
tiger_solved

## policy graph
policy_graph(tiger_solved)

## visualization
plot(tiger_solved)

library(igraph)

## use a different graph layout
plot(tiger_solved, layout = layout_with_mds)

## hide edge labels
plot(tiger_solved, edge.label = NA)

## custom larger vertex labels (A, B, ...)
plot(tiger_solved,
```

```
  vertex.label = LETTERS[1:nrow(solution(tiger_solved)$pg)],
  vertex.label.cex = 2,
  vertex.label.color = "white")

## add a plot title
plot(tiger_solved, main = model(tiger_solved)$name)
```

`policy_graph` *Extract the Policy Graph (as an igraph Object)*

Description

Convert the policy graph in a POMDP solution object into an igraph object.

Usage

```
policy_graph(x)
```

Arguments

x	A POMDP object.

Value

An object of class igraph containing a directed graph.

Author(s)

Hossein Kamalzadeh, Michael Hahsler

See Also

`solve_POMDP`

Examples

```
data("TigerProblem")
tiger_solved <- solve_POMDP(model = TigerProblem)
tiger_solved

policy_graph(tiger_solved)
```

POMDP *Define a POMDP Problem*

Description

Defines all the elements of a POMDP problem including discount rate, states, actions, observations, transition probabilities, observation probabilities, and rewards.

Usage

```
POMDP(discount, states, actions, observations, start, transition_prob,
      observation_prob, reward, values = "reward", name = NA)
```

Arguments

`discount` numeric, discount rate where 1 is 100%.

`states` character, a vector of strings specifying the names of the states.

`actions` character, a vector of strings specifying the names of the available actions.

`observations` character, a vector of strings specifying the names of the observations.

`start` Specifies the initial probabilities for each state. Options are:

- a probability distribution over the n states. That is, a vector of n probabilities, that add up to 1.
- the string `"uniform"` for a uniform distribution over all states.
- an integer in the range 1 to n to specify a single starting state. or
- a string specifying the name of a single starting state.
- a vector of strings, specifying a subset of states with a uniform start distribution. If the first element of the vector is `"-"`, then the following subset of states is excluded from the set of start states.

`transition_prob`

Specifies the transition probabilities between states. Options are:

- a data frame with 4 columns, where the columns specify *action*, *start-state*, *end-state* and the *probability* respectively. The first 3 columns could be either character (the name of the action or state) or integer indices.
- a named list of m (number of actions) matrices. Each matrix is square of size $n \times n$, where n is the number of states. The name of each matrix the action it applies to. Instead of a matrix, also the strings `"identity"` or `"uniform"` can be specified.

`observation_prob`

Specifies the observation probabilities. Options are:

- a data frame with 4 columns, where the columns specify *action*, *end-state*, *observation* and the *probability*, respectively. The first 3 columns could be either character (the name of the action, state, or observation), integer indices, or they can be `"*"` to indicate that the observation probability applies to all actions or states.

- a named list of m matrices, where m is the number of actions. Each matrix is of size $n \times o$, where n is the number of states and o is the number of observations. The name of each matrix is the action it applies to. Instead of a matrix, also the strings `"identity"` or `"uniform"` can be specified.

`reward` Specifies the rewards. Options are:

- a data frame with 5 columns, where the columns specify *action*, *start-state*, *end-state*, *observation* and the *reward*, respectively. The first 4 columns could be either character (names of the action, states, or observation), integer indices, or they can be `"*"` to indicate that the reward applies to all transitions.
- a named list of m lists, where m is the number of actions (names should be the actions). Each list contains n named matrices where each matrix is of size $n \times o$, in which n is the number of states and o is the number of observations. Names of these matrices should be the name of states.

`values` a string indicating if the specified rewards represent a `"reward"` or a `"cost"`. The default is reward.

`name` a string to identify the POMDP problem.

Details

POMDP problems can be solved using `solve_POMDP`. Details about the available specifications can be found in [1].

Value

The function returns an object of class POMDP_model which is list with the model specification.

Author(s)

Hossein Kamalzadeh, Michael Hahsler

References

[1] For further details on how the POMDP solver utilized in this R package works check the following website: http://www.pomdp.org

See Also

`solve_POMDP`

Examples

```
## The Tiger Problem

TigerProblem <- POMDP(
  name = "Tiger Problem",

  discount = 0.75,
```

```
  states = c("tiger-left" , "tiger-right"),
  actions = c("listen", "open-left", "open-right"),
  observations = c("tiger-left", "tiger-right"),

  start = "uniform",

  transition_prob = list(
    "listen" = "identity",
    "open-left" = "uniform",
    "open-right" = "uniform"),

  observation_prob = list(
    "listen" = matrix(c(0.85, 0.15, 0.15, 0.85), nrow = 2, byrow = TRUE),
    "open-left" = "uniform",
    "open-right" = "uniform"),

  reward = data.frame(
    "action" = c("listen", "open-left", "open-left", "open-right", "open-right"),
    "start-state" = c("*", "tiger-left", "tiger-right", "tiger-left", "tiger-right"),
    "end-state" = c("*", "*", "*", "*", "*"),
    "observation" = c("*", "*", "*", "*", "*"),
    "reward" = c(-1, -100, 10, 10, -100)),
  )

TigerProblem
```

`solution` *Extract the Solution of a POMDP*

Description

The function extracts the solution of a POMDP as an object of class POMDP_solution which is a list containing, e.g., the policy graph (pg) and the hyper-plane coefficients (alpha).

Usage

```
solution(x)
```

Arguments

x	object of class POMDP returned by `solve_POMDP`.

Value

returns an object is of class POMDP_solution, i.e., a list of all solution elements.

See Also

`solve_POMDP`

Examples

```
data("TigerProblem")
tiger_solved <- solve_POMDP(model = TigerProblem)
tiger_solved

solution(tiger_solved)
```

`solver_output` *Display the Output of the POMDP Solver*

Description

Displays the output generated by the solver 'pomdp-solve'. This includes used parameters, and iterations (i.e., epochs). This produces the same output as running `solve_POMDP` with the argument `verbose = TRUE`.

Usage

```
solver_output(x)
```

Arguments

`x`	object of class POMDP returned by `solve_POMDP`.

Value

returns invisibly a character string vector with the output of 'pomdp-solve'.

See Also

`solve_POMDP`

Examples

```
data("TigerProblem")
sol <- solve_POMDP(model = TigerProblem)

## solver output
solver_output(sol)
```

`solve_POMDP` *Solve a POMDP Problem*

Description

This function utilizes the 'pomdp-solve' program (written in C) to use different solution methods [2] to solve problems that are formulated as partially observable Markov decision processes (POMDPs) [1]. The result is a (close to) optimal policy.

Usage

```
solve_POMDP(model, horizon = NULL, method = "grid", parameter= NULL, verbose = FALSE)
solve_POMDP_parameter()
```

Arguments

`model`	a POMDP problem specification created with `POMDP`.
`method`	string; one of the following solution methods: `"grid"`, `"enum"`, `"twopass"`, `"witness"`, or `"incprune"`. Details can be found in [1].
`horizon`	an integer with the number of iterations for finite horizon problems. If set to `NULL`, the algorithm continues running iterations till it converges to the infinite horizon solution.
`parameter`	a list with parameters passed on to the pomdp-solve program.
`verbose`	logical, if set to `TRUE`, the function provides the output of the pomdp solver in the R console.

Details

`solve_POMDP_parameter()` displays available solver parameter options.

Value

The solver returns an object of class POMDP which is a list with the model specifications (`model`), the solution (`solution`), and the solver output (`solver_output`). The elements can be extracted with the functions `model`, `solution`, and `solver_output`.

Author(s)

Hossein Kamalzadeh, Michael Hahsler

References

[1] For further details on how the POMDP solver utilized in this R package works check the following website: http://www.pomdp.org

[2] Cassandra, A. Rocco, Exact and approximate algorithms for partially observable Markov decision processes, (1998). https://dl.acm.org/citation.cfm?id=926710

Examples

```
data("TigerProblem")

tiger_solved <- solve_POMDP(model = TigerProblem, parameter = list(fg_points = 10))
tiger_solved

## look at the model
model(tiger_solved)

## look at the solution
solution(tiger_solved)

## plot the policy graph
plot(tiger_solved)

## display solver options
solve_POMDP_parameter()
```

TigerProblem *Tiger Problem POMDP Specification*

Description

The model for the Tiger Problem [1].

Usage

```
data("TigerProblem")
```

Format

A list with the elements: discount, states, actions, observations, start, transition_prob, observation_prob, reward, name.

Details

The Tiger Problem is defined as follows [1]. A tiger is put with equal probability behind one of two doors, while treasure is put behind the other one. You are standing in front of the two closed doors and need to decide which one to open. If you open the door with the tiger, you will get hurt by the tiger (negative reward), but if you open the door with the treasure, you receive a positive reward. Instead of opening a door right away, you also have the option to wait and listen for tiger noises. But listening is neither free nor entirely accurate. You might hear the tiger behind the left door while it is actually behind the right door and vice versa.

The states of the system are tiger behind the left door (tiger-left) and tiger behind the right door (tiger-right).

Available actions are: open the left door (open-left), open the right door (open-right) or to listen (listen).

Rewards associated with these actions depend on the resulting state: +10 for opening the correct door (the door with treasure), -100 for opening the door with the tiger. A reward of -1 is the cost of listening.

As a result of listening, there are two observations: either you hear the tiger on the right (tiger-right), or you hear it on the left (tiger-left).

The transition probability matrix for the action listening is identity, i.e., the position of the tiger does not change. Opening either door means that the game restarts by placing the tiger uniformly behind one of the doors.

References

[1] Anthony R. Cassandra, Leslie P Kaelbling, and Michael L. Littman (1994). Acting Optimally in Partially Observable Stochastic Domains. In Proceedings of the Twelfth National Conference on Artificial Intelligence, pp. 1023-1028.

Examples

```
data(TigerProblem)

TigerProblem
```

`transitions` *Extract Transition Matrices from a POMDP*

Description

Extract the action-dependent transition probability matrices from a POMDP problem specification.

Usage

```
transitions(model)
```

Arguments

`model`	an object of class POMDP_model or POMDP.

Value

A list with state transition matrices.

Author(s)

Hossein Kamalzadeh, Michael Hahsler

See Also

POMDP

Examples

```
data("TigerProblem")

trans <- transitions(TigerProblem)
trans

## plot the Markov model for action listening (tiger stays in place)
library("igraph")

listen <- graph_from_adjacency_matrix(trans$listen, weighted = TRUE)
plot(listen, main = "Action: listen",
  layout = layout_on_grid, vertex.size = 40, edge.arrow.size =.5)
```

---

write_POMDP *Write a POMDP Model to a File in POMDP Format*

Description

Writes a POMDP file suitable for the pomdp-solve program. This function is used internally.

Usage

```
write_POMDP(model, file)
```

Arguments

model	an object of class POMDP_model.
file	a file name.

Author(s)

Hossein Kamalzadeh, Michael Hahsler

References

POMDP solver website: http://www.pomdp.org

See Also

POMDP

Index