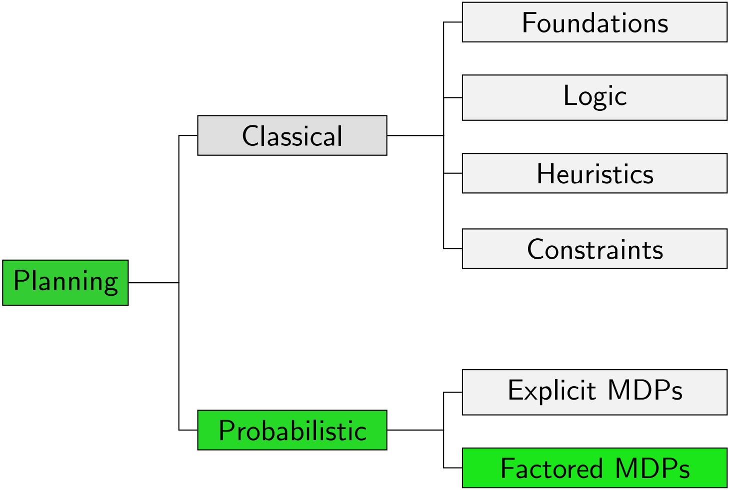
Planning and Optimization

G2. Real-time Dynamic Programming

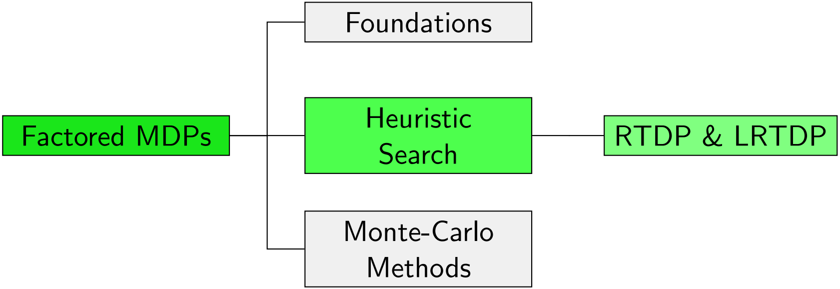
Malte Helmert and Gabriele Röger

Universität Basel

Content of this Course



Content of this Course: Factored MDPs



Motivation

Motivation: Real-time Dynamic Programming

- Asynchronous VI maintains table with state-value estimates for all states ...
- ... and has to update all states repeatedly.

Motivation: Real-time Dynamic Programming

- Asynchronous VI maintains table with state-value estimates for all states . . .
- . . . and has to update all states repeatedly.

- Real-time Dynamic Programming (RTDP) generates hash map with state-value estimates of relevant states
- uses admissible heuristic to achieve convergence albeit not updating all states
- Proposed by Barto, Bradtke & Singh (1995)

Real-time Dynamic Programming

Real-time Dynamic Programming

- RTDP updates only states relevant to the agent
- Originally motivated from agent that acts in environment by following greedy policy w.r.t. current state-value estimates.
- Performs Bellman backup in each encountered state
- Uses admissible heuristic for states not updated before

Trial-based Real-time Dynamic Programming

- We consider the offline version here.
  $\Rightarrow$ Interaction with environment is simulated in trials.
- In real world, outcome of action application cannot be chosen.
  $\Rightarrow$ In simulation, outcomes are sampled according to probabilities.

Real-time Dynamic Programming

RTDP for SSP $\mathcal{T} = \langle S, A, c, T, s_0, S_\star \rangle$

while more trials required:

$s := s_0$

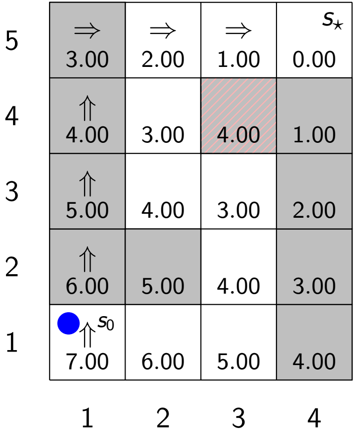
while $s \notin S_\star$:

$\hat{V}(s) := \min_{a \in A(s)} \left(c(a) + \sum_{s' \in S} T(s, a, s') \cdot \hat{V}(s') \right)$

$s :\sim \text{succ}(s, a_{\hat{V}}(s))$

Note: $\hat{V}(s)$ is maintained as a hash table of states. On the right hand side of line 4 or 5, if a state s is not in $\hat{V}$, $h(s)$ is used.

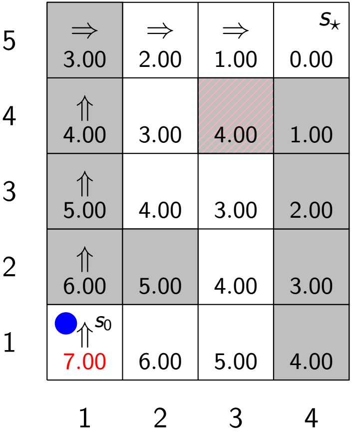
Example: RTDP



Start of 1st trial

Used heuristic: shortest path assuming agent never gets stuck

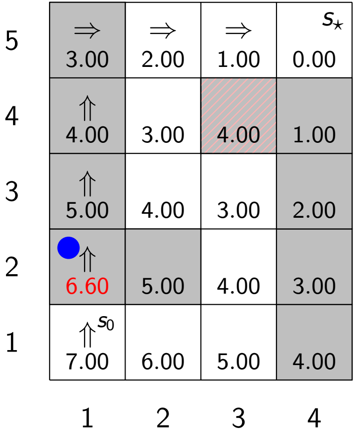
Example: RTDP



Step 1

Used heuristic: shortest path assuming agent never gets stuck

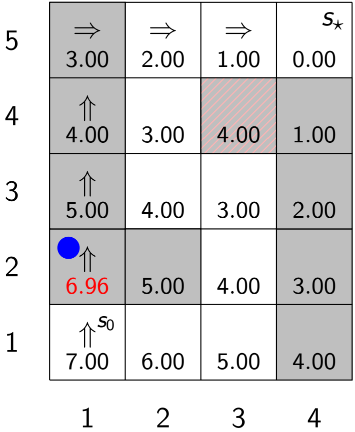
Example: RTDP



Step 2

Used heuristic: shortest path assuming agent never gets stuck

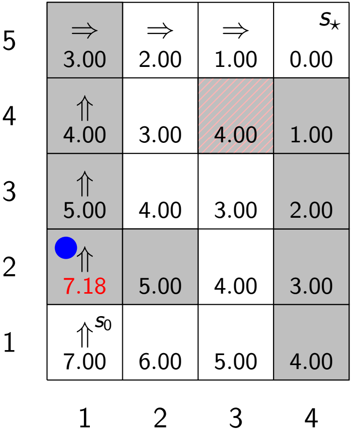
Example: RTDP



Step 3

Used heuristic: shortest path assuming agent never gets stuck

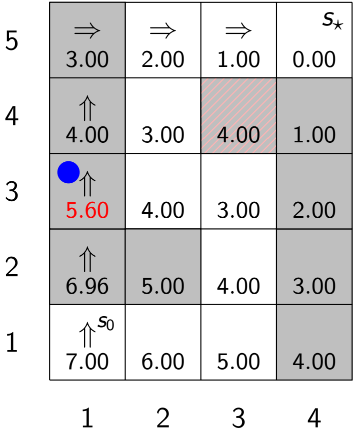
Example: RTDP



Step 4

Used heuristic: shortest path assuming agent never gets stuck

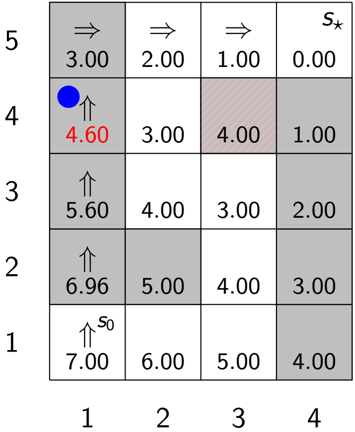
Example: RTDP



Step 5

Used heuristic: shortest path assuming agent never gets stuck

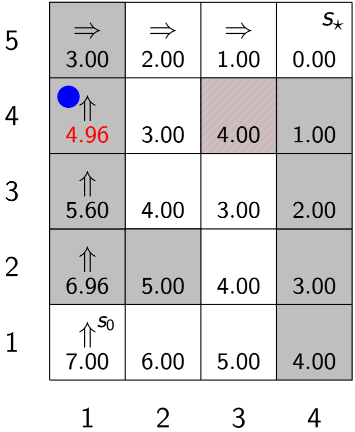
Example: RTDP



Step 6

Used heuristic: shortest path assuming agent never gets stuck

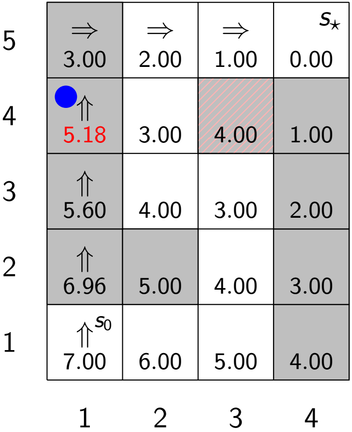
Example: RTDP



Step 7

Used heuristic: shortest path assuming agent never gets stuck

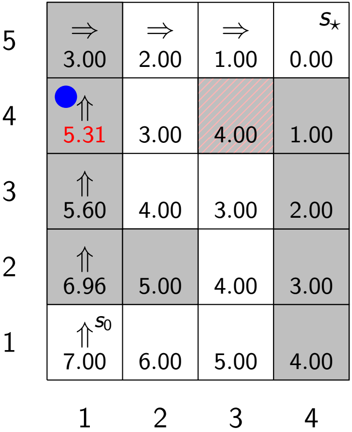
Example: RTDP



Step 8

Used heuristic: shortest path assuming agent never gets stuck

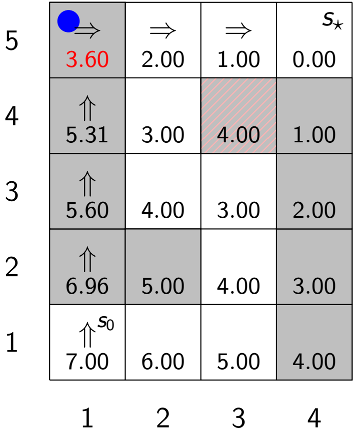
Example: RTDP



Step 9

Used heuristic: shortest path assuming agent never gets stuck

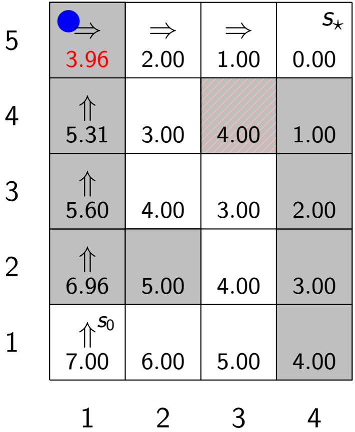
Example: RTDP



Step 10

Used heuristic: shortest path assuming agent never gets stuck

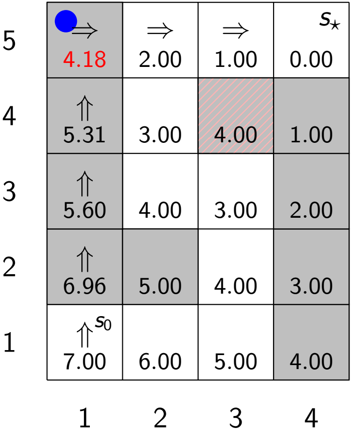
Example: RTDP



Step 11

Used heuristic: shortest path assuming agent never gets stuck

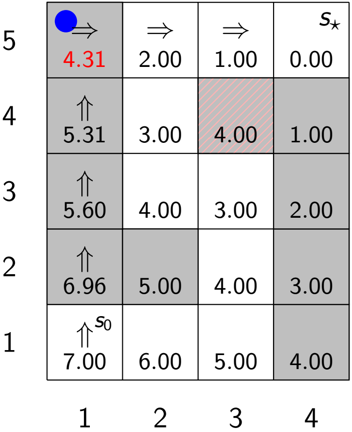
Example: RTDP



Step 12

Used heuristic: shortest path assuming agent never gets stuck

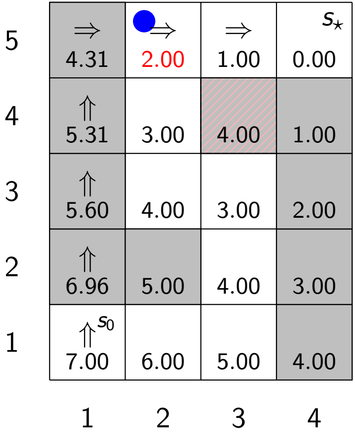
Example: RTDP



Step 13

Used heuristic: shortest path assuming agent never gets stuck

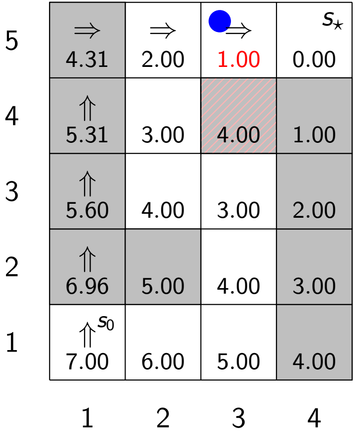
Example: RTDP



Step 14

Used heuristic: shortest path assuming agent never gets stuck

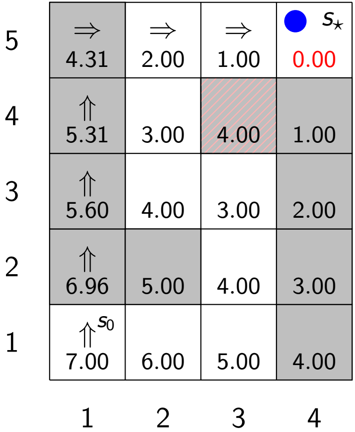
Example: RTDP



Step 15

Used heuristic: shortest path assuming agent never gets stuck

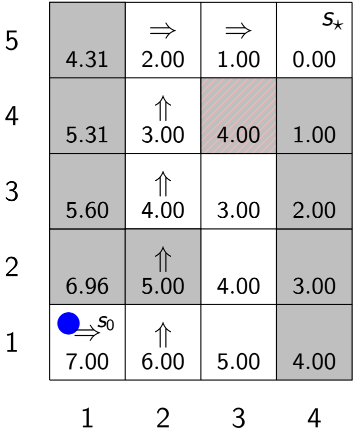
Example: RTDP



Step 16

Used heuristic: shortest path assuming agent never gets stuck

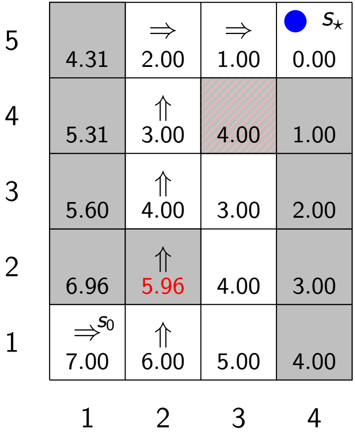
Example: RTDP



Start of 2nd trial

Used heuristic: shortest path assuming agent never gets stuck

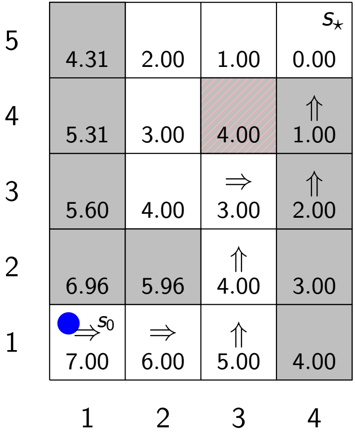
Example: RTDP



End of 2nd trial

Used heuristic: shortest path assuming agent never gets stuck

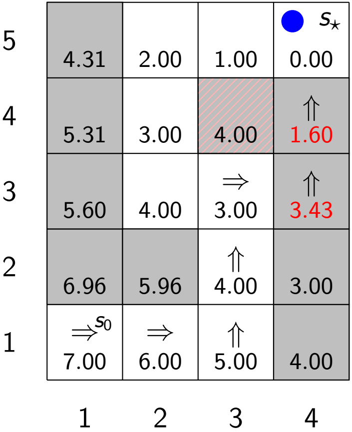
Example: RTDP



Start of 3rd trial

Used heuristic: shortest path assuming agent never gets stuck

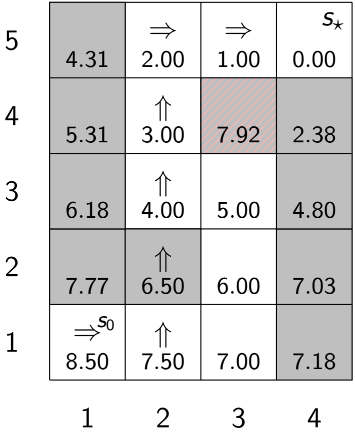
Example: RTDP



End of 3rd trial

Used heuristic: shortest path assuming agent never gets stuck

Example: RTDP



End of 16th trial

Used heuristic: shortest path assuming agent never gets stuck

RTDP: Theoretical Properties

Theorem

Using an admissible heuristic, RTDP converges to an optimal solution without (necessarily) computing state-value estimates for all states.

Proof omitted.

Labeled Real-time Dynamic Programming

Motivation

Issues of RTDP:

- States are still updated after state-value estimate has converged.
- No termination criterion $\Rightarrow$ algorithm is underspecified

Most popular algorithm to overcome these shortcomings:
Labeled RTDP (Bonet & Geffner, 2003)

Labeled RTDP: Idea

The main idea of Labeled RDTP (LRTDP) is to label states as solved

- Each trial terminates when a solved state is encountered
  $\Rightarrow$ solved states no longer updated
- LRTDP terminates when the initial state is labeled as solved
  $\Rightarrow$ well-defined termination criterion

Solved States in SSPs

- States are solved if the state-value estimate changes only little
- In presence of cycles, all states in a strongly connected component (SCC) are considered simultaneously
- Labeled RTDP uses sub-algorithm `CheckSolved` to check whether all states in a SCC are solved

CheckSolved Procedure

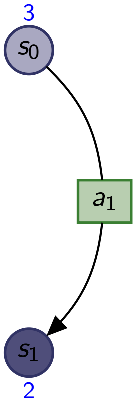
- `CheckSolved` is called on all states that were encountered in a trial in reverse order.
- `CheckSolved` checks how much the state-value estimates of unlabeled states reachable under the greedy policy would change with another update.
- If this change is below some constant ε for all these states then they are all labeled as solved.
- Otherwise, `CheckSolved` performs an additional backup for the encountered states, hence improving the state value estimate for at least one of them.

Labeled RTDP: Example ($\varepsilon = 0.005$)

visited: s_0

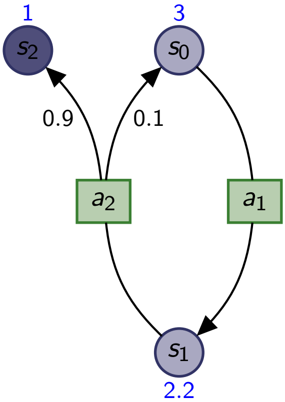
Labeled RTDP: Example ($\varepsilon = 0.005$)

visited: s_0, s_1

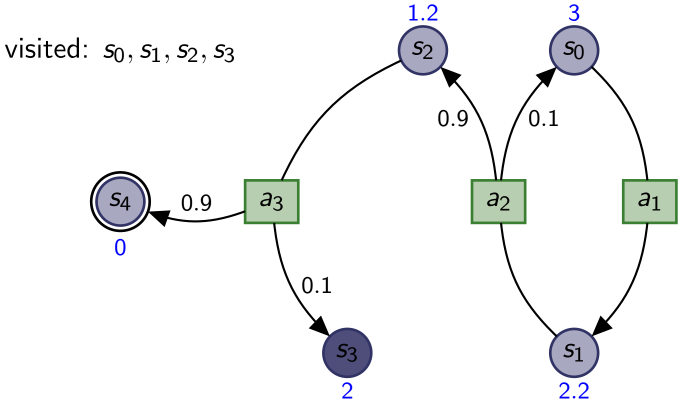


Labeled RTDP: Example ($\varepsilon = 0.005$)

visited: s_0, s_1, s_2

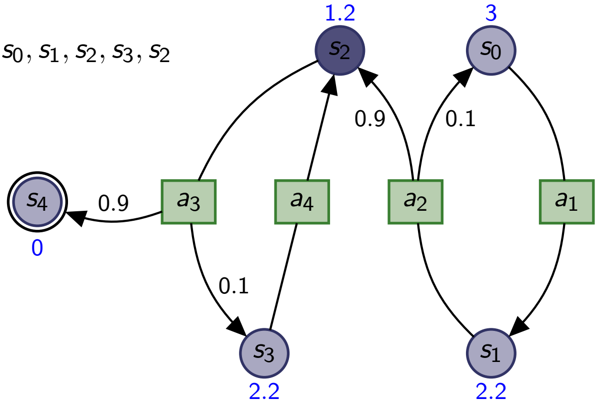


Labeled RTDP: Example ($\varepsilon = 0.005$)



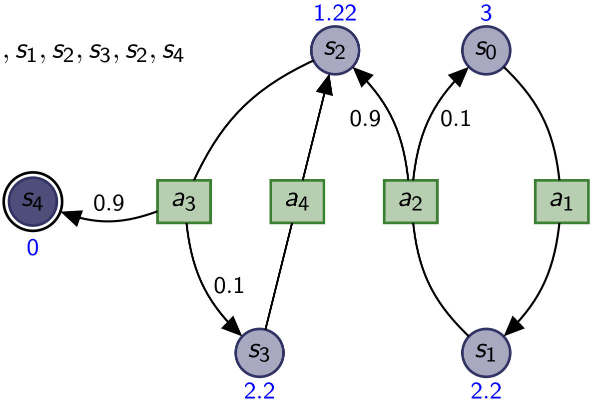
Labeled RTDP: Example ($\varepsilon = 0.005$)

visited: s_0, s_1, s_2, s_3, s_2



Labeled RTDP: Example ($\varepsilon = 0.005$)

visited: $s_0, s_1, s_2, s_3, s_2, s_4$

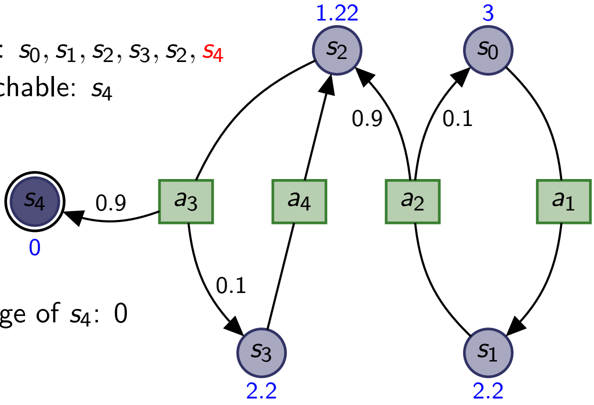


Labeled RTDP: Example ($\varepsilon = 0.005$)

check_solved: $s_0, s_1, s_2, s_3, s_2, s_4$

reachable: s_4

change of s_4: 0



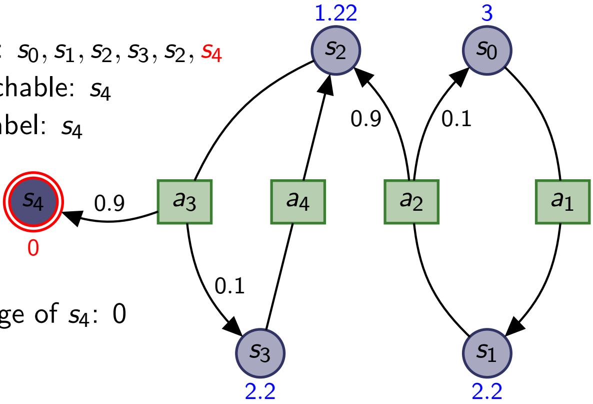
Labeled RTDP: Example ($\varepsilon = 0.005$)

check_solved: $s_0, s_1, s_2, s_3, s_2, s_4$

reachable: s_4

label: s_4

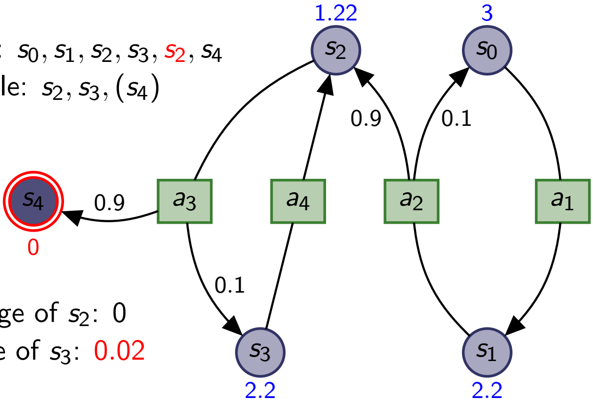
change of s_4: 0



Labeled RTDP: Example ($\varepsilon = 0.005$)

check_solved: $s_0, s_1, s_2, s_3,$ s_2, s_4

reachable: $s_2, s_3, (s_4)$



change of s_2: 0

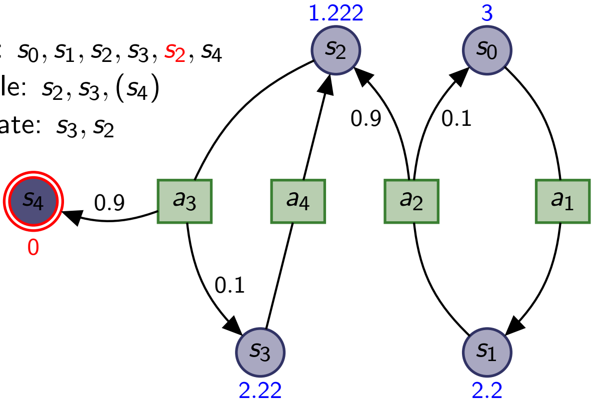
change of s_3: 0.02

Labeled RTDP: Example ($\varepsilon = 0.005$)

check_solved: $s_0, s_1, s_2, s_3, s_2, s_4$

reachable: $s_2, s_3, (s_4)$

update: s_3, s_2



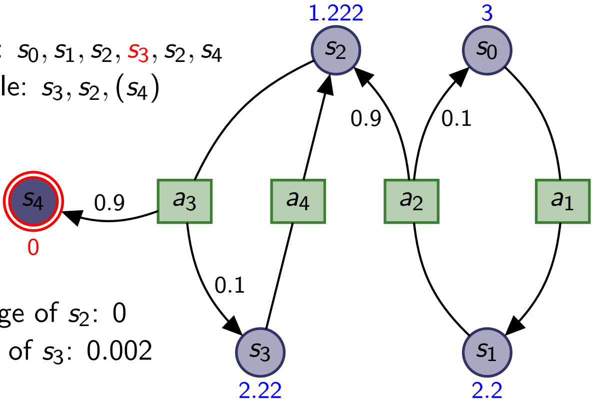
Labeled RTDP: Example ($\varepsilon = 0.005$)

check_solved: $s_0, s_1, s_2, s_3, s_2, s_4$

reachable: $s_3, s_2, (s_4)$

change of s_2: 0

change of s_3: 0.002

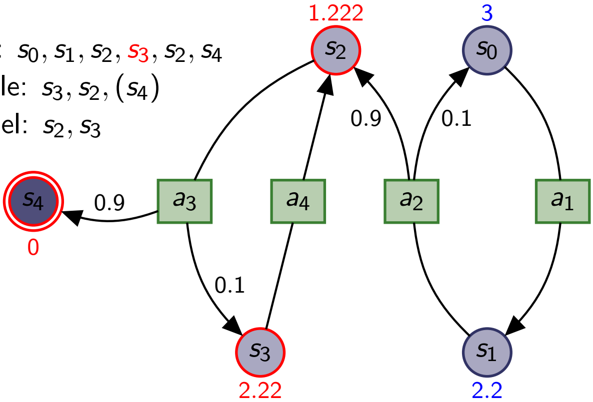


Labeled RTDP: Example ($\varepsilon = 0.005$)

check_solved: $s_0, s_1, s_2, s_3, s_2, s_4$

reachable: $s_3, s_2, (s_4)$

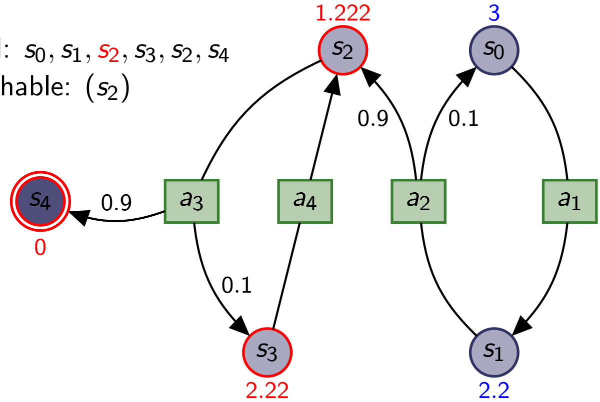
label: s_2, s_3



Labeled RTDP: Example ($\varepsilon = 0.005$)

check_solved: $s_0, s_1, s_2, s_3, s_2, s_4$

reachable: (s_2)



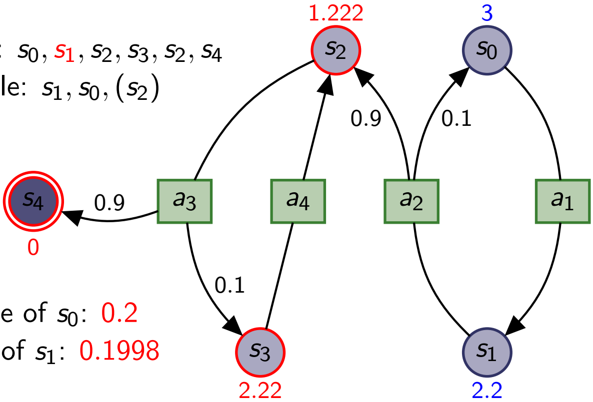
Labeled RTDP: Example ($\varepsilon = 0.005$)

check_solved: $s_0, s_1, s_2, s_3, s_2, s_4$

reachable: $s_1, s_0, (s_2)$

change of s_0: 0.2

change of s_1: 0.1998

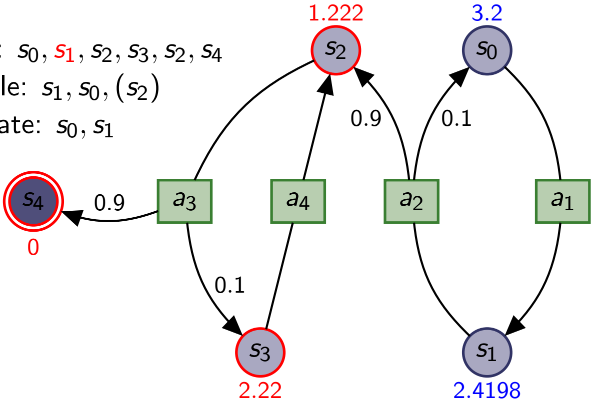


Labeled RTDP: Example ($\varepsilon = 0.005$)

check_solved: $s_0, s_1, s_2, s_3, s_2, s_4$

reachable: $s_1, s_0, (s_2)$

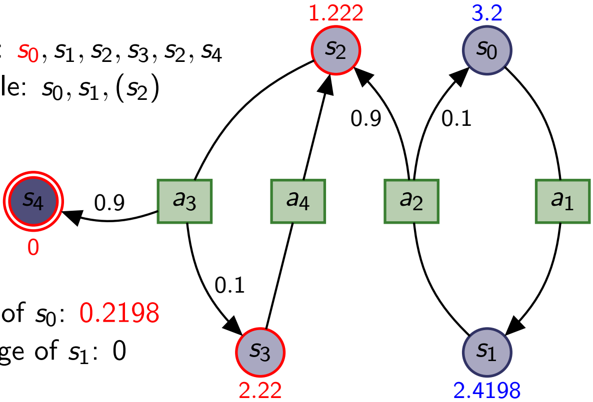
update: s_0, s_1



Labeled RTDP: Example ($\varepsilon = 0.005$)

check_solved: $s_0, s_1, s_2, s_3, s_2, s_4$

reachable: $s_0, s_1, (s_2)$



change of s_0: 0.2198

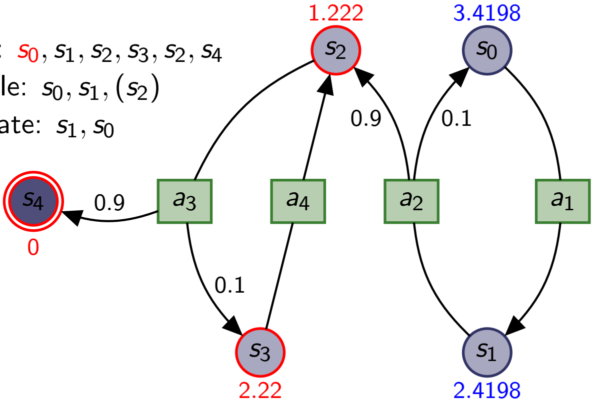
change of s_1: 0

Labeled RTDP: Example ($\varepsilon = 0.005$)

check_solved: $s_0, s_1, s_2, s_3, s_2, s_4$

reachable: $s_0, s_1, (s_2)$

update: s_1, s_0



Labeled Real-time Dynamic Programming

Labeled RTDP for SSP $\mathcal{T}$

while s_0 is not solved:

  visit(s_0)

visit state s

if s is solved or $s \in S_\star$:

  return

$\hat{V}(s) := \min_{a \in A(s)} \left(c(a) + \sum_{s' \in S} T(s, a, s') \cdot \hat{V}(s') \right)$

$s' :\sim \text{succ}(s, a_{\hat{V}}(s))$

visit(s')

check_solved(s)

$\hat{V}(s)$ is maintained as a hash table of states. On the right hand side of line 3 or 4 in `visit`(s), if a state s is not in $\hat{V}$, $h(s)$ is used.

Labeled RTDP: CheckSolved

check_solved for state s

set allsolved := true, open, closed := stack
if s not labeled **then** push s to open
while open is not empty:
    pop s' from open and insert it into closed
    if change of $s' > \varepsilon$
        allsolved := false
    else push all $s'' \in \text{succ}(s', a_{\hat{V}}(s'))$ to open that are
        not labeled and not in open or closed
if allsolved **then** label all states in closed as solved
else
    while closed is not empty:
        pop s' from closed and update its state value

Labeled RTDP: Theoretical Properties

Theorem

Using an admissible heuristic, Labeled RTDP converges to an optimal solution without (necessarily) computing state-value estimates for all states.

Proof omitted.

Experimental Results [Bonet and Geffner, ICAPS 2003]

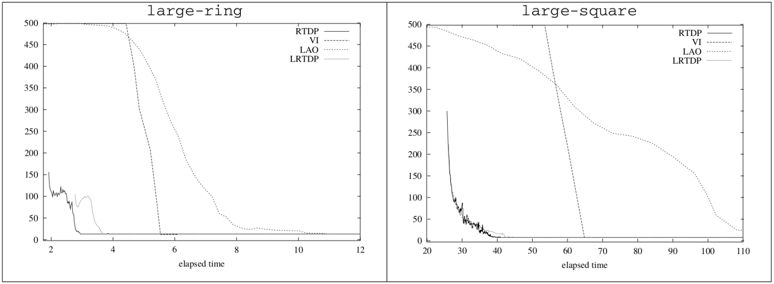


Figure 3: Quality profiles: Average cost to the goal vs. time for RTDP, VI, ILAO* and LRTDP with the heuristic $h = 0$ and $\epsilon = 10^{-3}$.

algorithm	small-b	large-b	h-track	small-r	large-r	small-s	large-s	small-y	large-y
VI($h = 0$)	1.101	**4.045**	**15.451**	0.662	5.435	5.896	78.720	16.418	61.773
ILAO*($h = 0$)	2.568	11.794	43.591	1.114	11.166	12.212	250.739	57.488	182.649
LRTDP($h = 0$)	**0.885**	7.116	15.591	**0.431**	**4.275**	**3.238**	**49.312**	**9.393**	**34.100**

Table 2: Convergence time in seconds for the different algorithms with initial value function $h = 0$ and $\epsilon = 10^{-3}$. Times for RTDP not shown as they exceed the cutoff time for convergence (10 minutes). Faster times are shown in bold font.

algorithm	small-b	large-b	h-track	small-r	large-r	small-s	large-s	small-y	large-y
VI(h_{min})	1.317	4.093	12.693	0.737	5.932	6.855	102.946	17.636	66.253
ILAO*(h_{min})	1.161	2.910	11.401	0.309	3.514	0.387	1.055	0.692	1.367
LRTDP(h_{min})	**0.521**	**2.660**	**7.944**	**0.187**	**1.599**	**0.259**	**0.653**	**0.336**	**0.749**

Table 3: Convergence time in seconds for the different algorithms with initial value function $h = h_{min}$ and $\epsilon = 10^{-3}$. Times for RTDP not shown as they exceed the cutoff time for convergence (10 minutes). Faster times are shown in bold font.

Summary

Summary

- Real-time Dynamic Programming is an optimal algorithm for SSPs ...
- ... that backups only a subset of states ...
- ... without generating an explicit representation of the state-space.
- Labeled RTDP labels states as solved to stop updating converged states ...
- ... and speeds up convergence with additional backups in reverse order.