Planning and Optimization

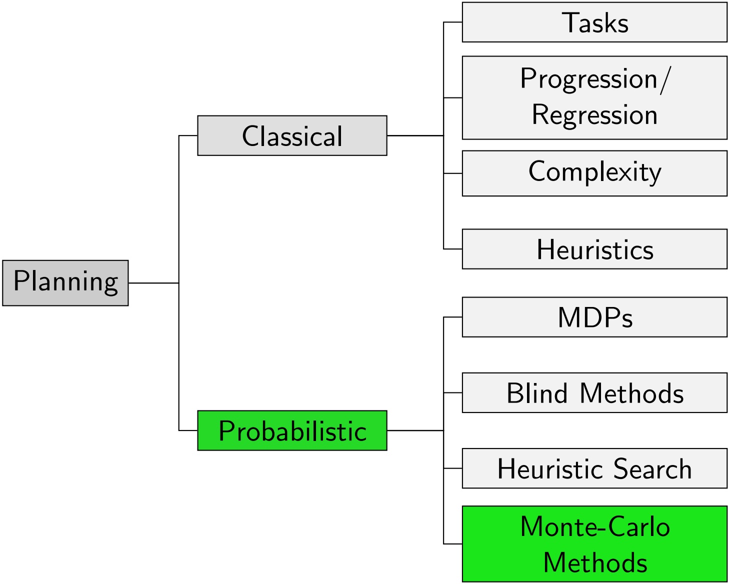
G8. Trial-based Heuristic Tree Search

Gabriele Röger and Thomas Keller

Universität Basel

December 17, 2018

Content of this Course



Motivation

AO* & LAO*: Recap

- Iteratively build explicated graph
- Extend explicated graph by expanding fringe node in partial solution graph
- State-value estimates are initialized with admissible heuristic
- Propagate information with Bellman backups in partial solution graph

(Labeled) Real-Time Dynamic Programming: Recap

- Iteratively performs trials
- Simulates greedy policy in each trial
- Encountered states are updated with Bellman backup
- Admissible heuristic used if no state-value estimate available
- Labeling procedure marks states that have converged

Monte-Carlo Tree Search: Recap

- Iteratively explicates search tree in trials
- Uses tree policy to traverse tree
- First encountered state not yet in tree added to search tree
- State-value estimates are initialized with default policy
- Propagates information with Monte-Carlo backups in reverse order through visited states

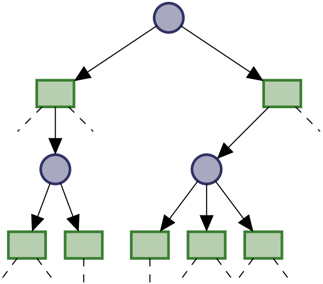
Trial-based Heuristic Tree Search

- All are asymptotically optimal (or such a version exists)
- In practice, all have complementary strengths
- There are a significant differences between these algorithms
- but they also have a lot in common
- common framework that allows to describe all three: Trial-based Heuristic Tree Search (THTS)

Trial-based Heuristic Tree Search Framework

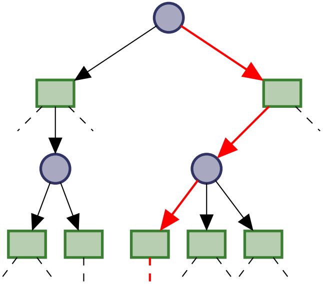
Trial-based Heuristic Tree Search

- Perform trials to explicate search tree
  - decision (OR) nodes for states
  - chance (AND) nodes for actions
- Annotate nodes with
  - state-/action-value estimate
  - visit counter
  - solved label
- Initialize search nodes with heuristic



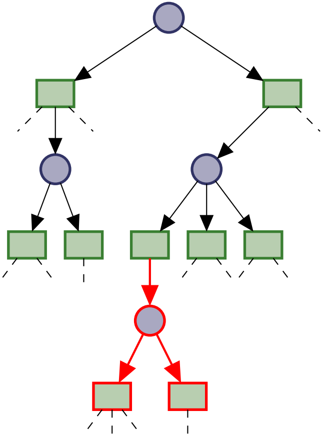
Trial-based Heuristic Tree Search

- Perform trials to explicate search tree
  - decision (OR) nodes for states
  - chance (AND) nodes for actions
- Annotate nodes with
  - state-/action-value estimate
  - visit counter
  - solved label
- Initialize search nodes with heuristic
- 6 variable ingredients:
  - action selection
  - outcome selection



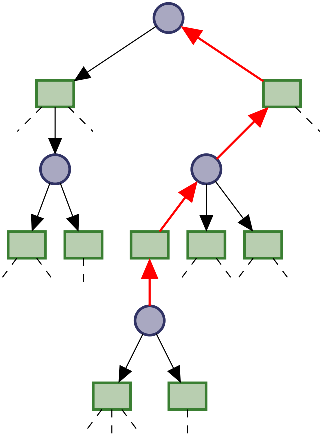
Trial-based Heuristic Tree Search

- Perform trials to explicate search tree
  - decision (OR) nodes for states
  - chance (AND) nodes for actions
- Annotate nodes with
  - state-/action-value estimate
  - visit counter
  - solved label
- Initialize search nodes with heuristic
- 6 variable ingredients:
  - action selection
  - outcome selection
  - initialization
  - trial length



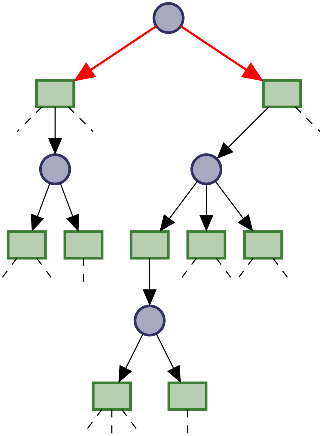
Trial-based Heuristic Tree Search

- Perform trials to explicate search tree
  - decision (OR) nodes for states
  - chance (AND) nodes for actions
- Annotate nodes with
  - state-/action-value estimate
  - visit counter
  - solved label
- Initialize search nodes with heuristic
- 6 variable ingredients:
  - action selection
  - outcome selection
  - initialization
  - trial length
  - backup function



Trial-based Heuristic Tree Search

- Perform trials to explicate search tree
  - decision (OR) nodes for states
  - chance (AND) nodes for actions
- Annotate nodes with
  - state-/action-value estimate
  - visit counter
  - solved label
- Initialize search nodes with heuristic
- 6 variable ingredients:
  - action selection
  - outcome selection
  - initialization
  - trial length
  - backup function
  - recommendation function



Trial-based Heuristic Tree Search

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

d_0 = create root node associated with s_0
while time allows:
    visit_decision_node($d_0, \mathcal{T}$)
return recommend(d_0)

THTS: Visit a Decision Node

visit_decision_node for decision node d, SSP $\mathcal{T} = \langle S, L, c, T, s_0, S_\star \rangle$

if $s(d) \in S_\star$ **then return** 0
$a :=$ select_action(d)
if a not explicated:
    cost = expand_and_initialize(d, a)
if not trial_length_reached(d)
    let c be the node in children(d) with $a(c) = a$
    cost = visit_chance_node($c, \mathcal{T}$)
backup(d,cost)
return cost

THTS: Visit a Chance Node

visit_chance_node for chance node c, SSP $\mathcal{T} = \langle S, L, c, T, s_0, S_\star \rangle$

$s' =$ select_outcome$(s(c), a(c))$
if s' not explicated:
    cost $=$ expand_and_initialize(c, s')
if not trial_length_reached(c)
    let d be the node in children(c) with $s(d) = s'$
    cost $=$ visit_decision_node$(d, \mathcal{T})$
cost $=$ cost $+ \, c(s(c), a(c))$
backup$(c$,cost$)$
return cost

THTS Algorithms

MCTS in the THTS Framework

- Trial length: terminate trial when node is explicated
- Action selection: tree policy
- Outcome selection: sample
- Initialization: add single node to the tree
  and initialize with heuristic that simulates the default policy
- Backup function: Monte-Carlo backups
- Recommendation function: expected best arm

AO* (Tree Search Version) in the THTS Framework

- Trial length: terminate trial when node is expanded
- Action selection: greedy
- Outcome selection: depends on AO* version
- Initialization: expand decision node and all its chance node successors, then initialize all $\hat{V}^k$ with admissible heuristic
- Backup function: Bellman backups & solved labels
- Recommendation function: expected best arm

LRTDP (Tree Search Version) in the THTS Framework

- Trial length: finish trials only in goal states
- Action selection: greedy
- Outcome selection: sample unsolved outcome
- Initialization: expand decision node and all its chance node successors, then initialize all $\hat{V}^k$ with admissible heuristic
- Backup function: Bellman backups & solved labels
- Recommendation function: expected best arm

Further Ingredients from Literature

- Recommendation function:
  - Most played arm [Bubeck et al. 2009, Chaslot et al. 2008]
  - Empirical distribution of plays [Bubeck et al. 2009]
  - Secure arm [Chaslot et al. 2008]
- Initialization:
  - Expand decision node and initialize chance nodes with heuristic for state-action pairs [Keller & Eyerich, 2012]
  - Any classical heuristic on any determinization
  - Occupation measure heuristic [Trevizan et al., 2017]

Further Ingredients from Literature

Backup functions:

- Temporal Differences [Sutton & Barto, 1987]
- Q-Learning [Watkins, 1989]
- Selective Backups [Feldman & Domshlak, 2012; Keller, 2015]
- MaxMonte-Carlo [Keller & Helmert, 2013]
- Partial Bellman [Keller & Helmert, 2013]

Further Ingredients from Literature

Action selections:

- Uniform sampling (UNI)
- ε-greedy (ε-G)
- ε-G with decaying ε:
  - ε_{LIN}-G [Singh et al., 2000; Auer et al., 2002]
  - ε_{RT}-G [Keller, 2015]
  - ε_{LOG}-G [Keller, 2015]
- Boltzmann exploration (BE)
- BE with logarithmic decaying τ (BE-DT) [Singh et al., 2000]
- UCB1 [Auer et al., 2002]
- Root-valued UCB (RT-UCB) [Keller, 2015]

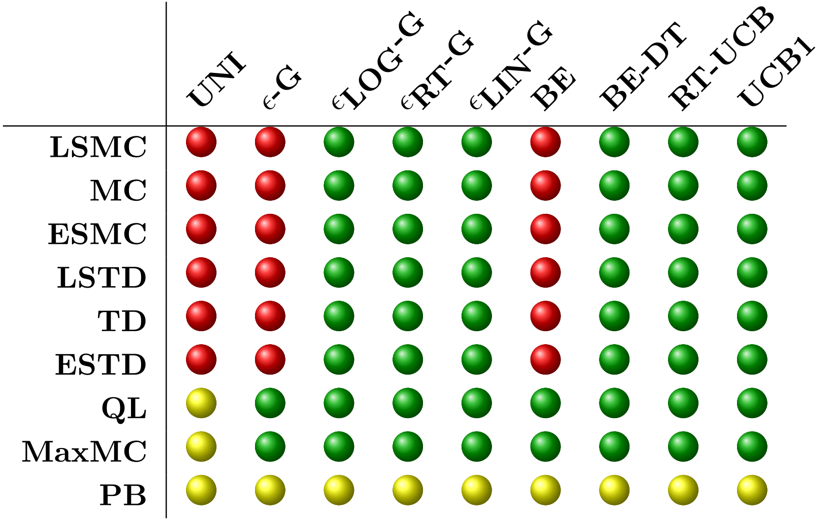
Experimental Comparison

- THTS allows to mix and match ingredients
- Not all combinations asymptotically optimal
- Analysis based on properties of ingredients possible

Experimental Comparison

- THTS allows to mix and match ingredients
- Not all combinations asymptotically optimal
- Analysis based on properties of ingredients possible
- In [Keller, 2015], comparison of:
  - 1 trial length, 1 outcome selection, 1 initialization
  - 2 different recommendation functions
  - 9 different backup functions
  - 9 different action selections
- $\Rightarrow$ 162 different THTS algorithms
- 115 shown to be asymptotically optimal

Asymptotic Optimality



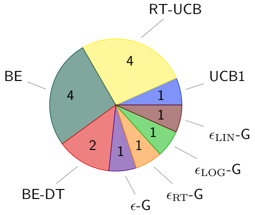
Experimental Evaluation

- Most played arm recommendation function often better than same configuration with expected best arm

	Academic	Crossing	Elevators	Game	Navigation	Recon	Skill	Sysadmin	Tamarisk	Traffic	Triangle	Wildfire	**Total**
MC^{UCB1}_{MPA}	27	65	78	86	45	92	77	89	86	71	46	84	**70**
Prost 2011	26	62	49	84	42	90	69	88	83	60	49	85	**66**

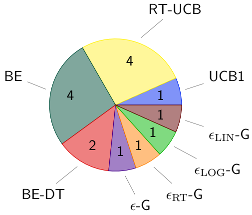
Experimental Evaluation

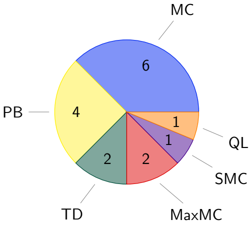
- Most played arm recommendation function often better than same configuration with expected best arm
- Boltzman exploration and root-valued UCB1 perform best in most domains



Experimental Evaluation

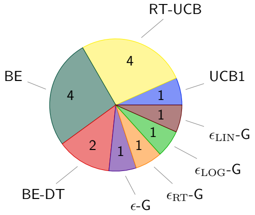
- Most played arm recommendation function often better than same configuration with expected best arm
- Boltzman exploration and root-valued UCB1 perform best in most domains
- Monte-Carlo and Partial Bellman backups perform best in most domains

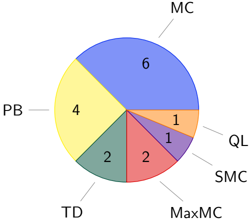




Experimental Evaluation

- Most played arm recommendation function often better than same configuration with expected best arm
- Boltzman exploration and root-valued UCB1 perform best in most domains
- Monte-Carlo and Partial Bellman backups perform best in most domains
- almost all action selections and backup functions perform best in at least one domain





Implementation: Prost

- The Prost planner implements THTS framework
- mixing and matching of ingredients very simple
- to add new ingredients, just inherit from the corresponding class

`https://bitbucket.org/tkeller/prost/`

Summary

Summary

- MCTS, AO*and RTDP have complementary strengths
- But also a similar structure
- THTS allows to combine ideas from MCTS, Heuristic Search and DP
- Mixing and matching ingredients leads to novel and sometimes better algorithms