Metareasoning for Deliberation Time Distribution in the PROST Planner

Ferdinand Badenberg

University of Basel

Bachelor Thesis Presentation, 2017

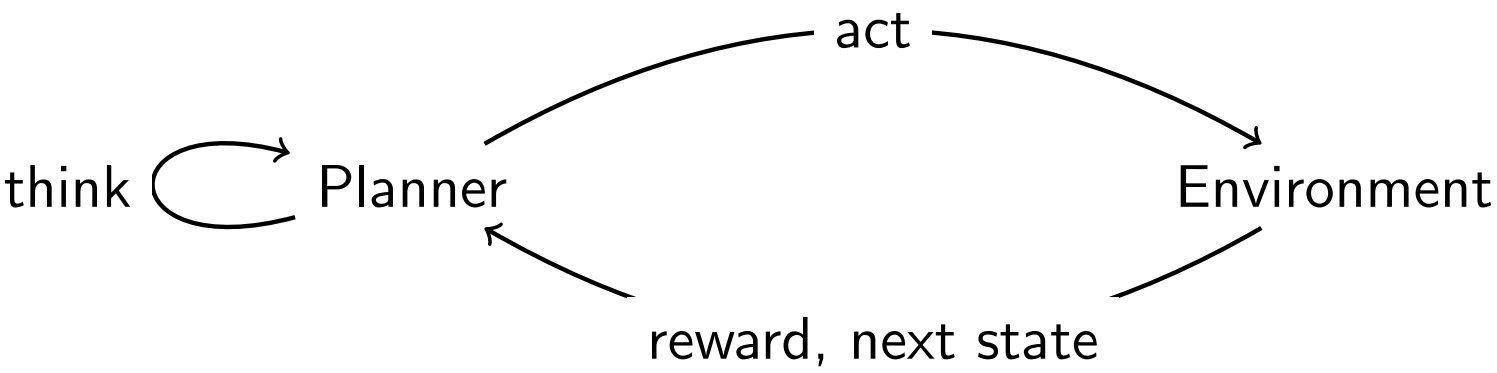
Outline

1. Motivation
   - Why Metareasoning?
   - Metareasoning Problem
2. Approaches for Metareasoners
   - Hand Made Functions
   - Metareasoner of Lin. et al.
   - Improvements for the Metareasoner
3. Results
   - Results for the Hand Made Functions
   - Results for the Formal Procedure

Table of Contents

1. Motivation
   - Why Metareasoning?
   - Metareasoning Problem
2. Approaches for Metareasoners
   - Hand Made Functions
   - Metareasoner of Lin. et al.
   - Improvements for the Metareasoner
3. Results
   - Results for the Hand Made Functions
   - Results for the Formal Procedure

Cycle



Motivation

Why Metareasoning?

- Optimise policy in given time
- Allocate time to think where it is needed
- Act if decision is easy, clear best action
- Think if decision is difficult, multiple actions very close

Metareasoning Problem

Metareasoning Problem

- Steps from finite horizon MDP
- Rounds
- Limited time
- Anytime search algorithm

Metareasoner

Decision to think or act

- Based on specific values for these factors
- After one thinking cycle of the algorithm
- Goal: only think when necessary

Table of Contents

1 Motivation
- Why Metareasoning?
- Metareasoning Problem

2 Approaches for Metareasoners
- Hand Made Functions
- Metareasoner of Lin. et al.
- Improvements for the Metareasoner

3 Results
- Results for the Hand Made Functions
- Results for the Formal Procedure

Hand Made Functions

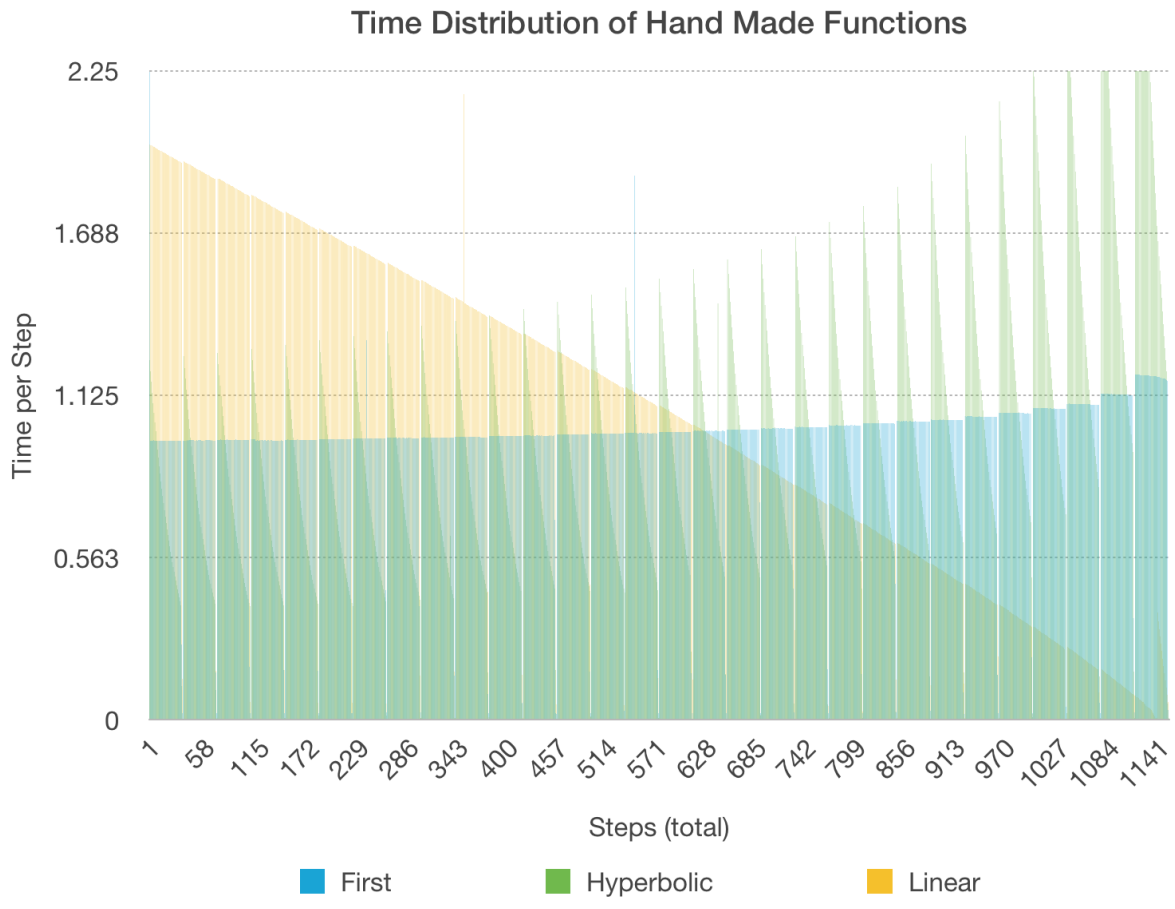
Idea

- Allocate time for each step
- Think for as long as time is left
- State of the search algorithm not considered

Functions Tested:

1. Uniform (Standard)
2. First
3. Linear
4. Hyperbolic

Example Time Distribution



Formal Metareasoner of Lin et al.

Metareasoner

- Idea: think if change of policy is likely, act if it will stay the same
- Only considers expected reward estimations (Q-values) of search algorithm
- Act if $Q^{act} \geq Q^{think}$
- How are they calculated?

Formal Metareasoner: Q^{think} and Q^{act}

Q^{think}

- Expected reward of the policy after another thinking cycle
- Simplification: only best action is relevant
- Estimate probability of action a being the best after the next thinking cycle
- Estimate expected reward given that action a is chosen
- Needed: next Q-values for each action

Q^{act}

- Intuitive idea: Q-value of current best action
- But: average of current Q-value and next Q-value

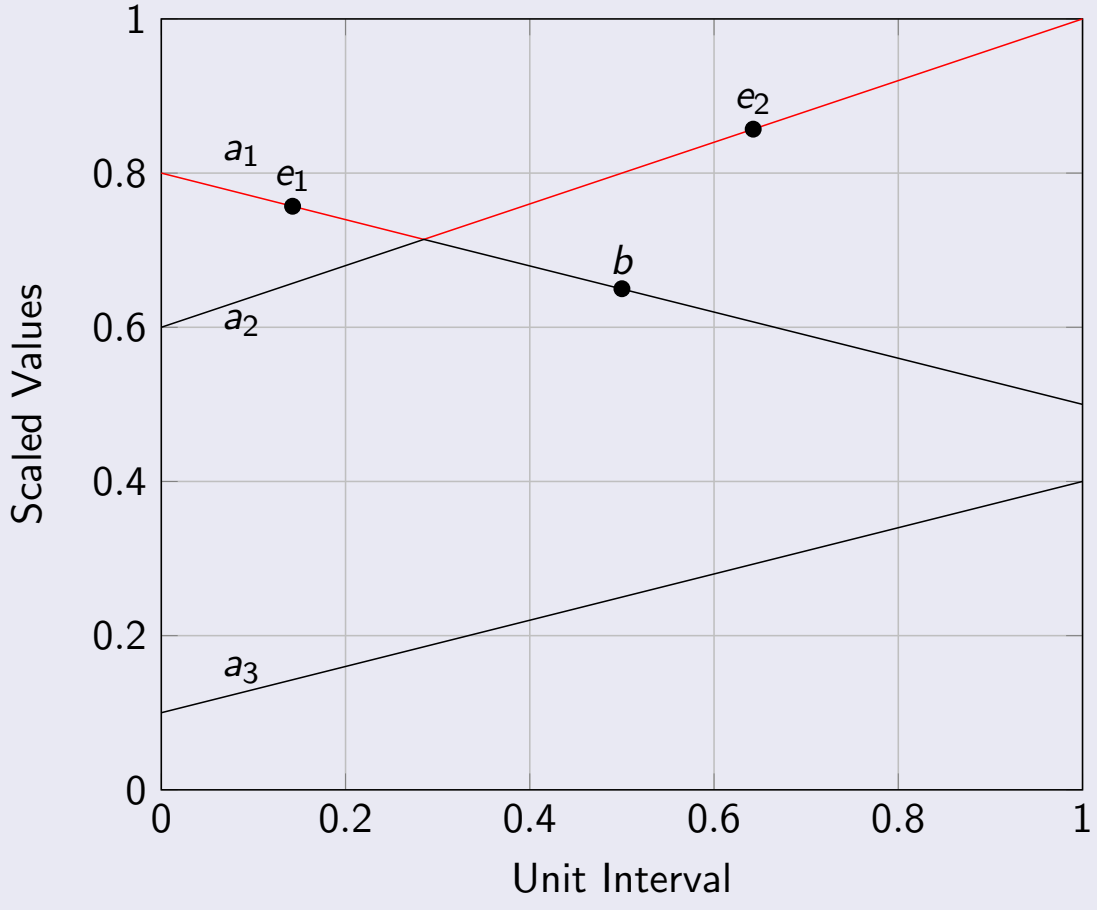
Formal Metareasoner: Estimation of Next Q-values

Estimating Next Q-values

- Idea: base next change in Q-values on previous change in Q-values
- Assumption: next ΔQ-value no larger than the previous one
- Draw random ρ between 0 and 1
- $\Delta Q(a) = \hat{\Delta} Q(a) * \rho$ for all actions a

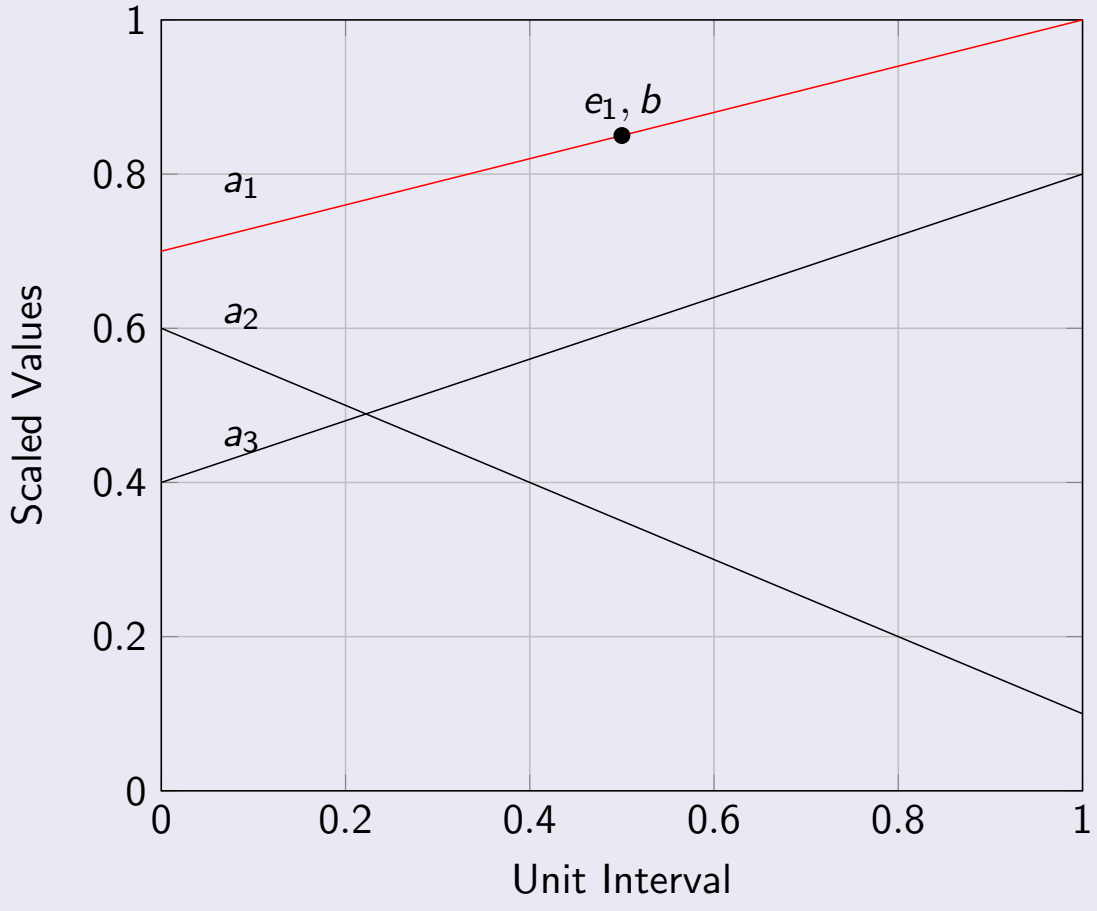
Line Segment Example: UCT

$Q^{think} > Q^{act}$



Line Segment Example: UCT

$Q^{think} = Q^{act}$



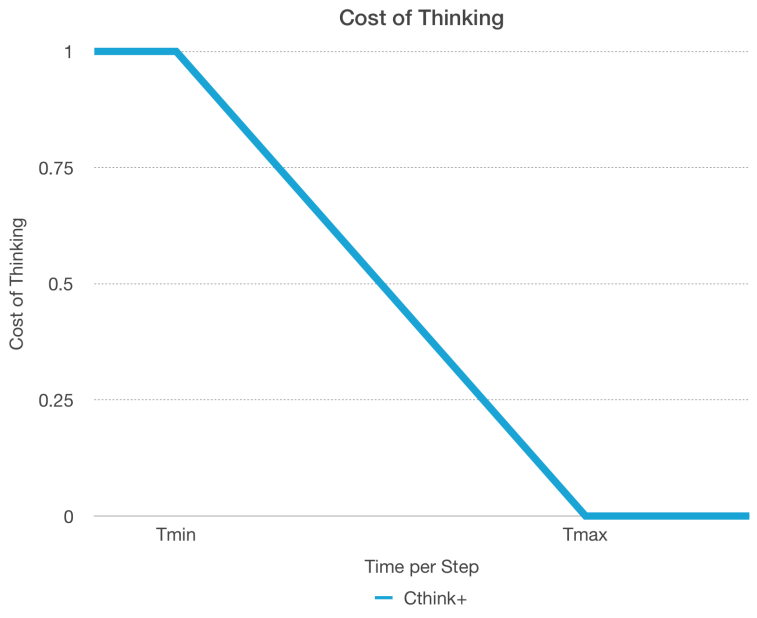
Improvements

Minimum Thinking Time

Problem: assumption is often not true early on

Improvement: think for at least T_{min} seconds

Cthink$^+$

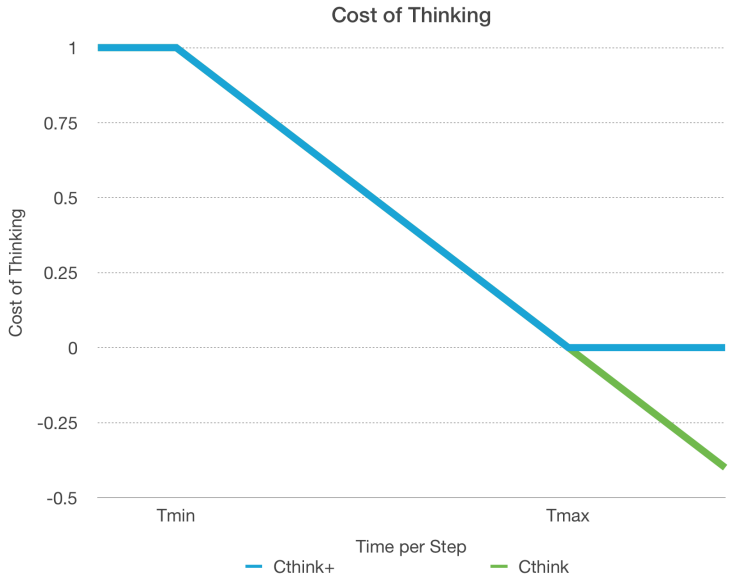


Cthink$^+$

Problem: time left is not considered

Improvement: subtract C^{think} from Q^{think}

Cthink



Cthink

Problem: stopping with time left is useless

Improvement: allow a negative C^{think}

Table of Contents

1. Motivation
   - Why Metareasoning?
   - Metareasoning Problem
2. Approaches for Metareasoners
   - Hand Made Functions
   - Metareasoner of Lin. et al.
   - Improvements for the Metareasoner
3. Results
   - Results for the Hand Made Functions
   - Results for the Formal Procedure

Results Hand Made Functions

Results

Problem	Uniform	Hyperbolic	First	Linear
Wildfire	74	71	80	**81**
Triangle	72	65	72	**75**
Academic	37	37	34	**45**
Elevators	93	93	91	**94**
Tamarisk	93	**94**	92	91
Sysadmin	**94**	**94**	90	91
Recon	97	**99**	97	96
Game	**97**	93	94	93
Traffic	**97**	**97**	96	96
Crossing	87	89	91	**99**
Skill	91	91	88	**93**
Navigation	65	58	**83**	82
Total	83	82	84	**86**

Results Formal Procedure

Results

Problem	Uniform	Lin et al.	Minimum	Cthink$^+$	Cthink
Wildfire	60	90	86	**95**	68
Triangle	**78**	67	62	59	68
Academic	**39**	32	36	35	38
Elevators	**98**	71	83	83	97
Tamarisk	**96**	68	86	90	92
Sysadmin	**100**	36	67	74	82
Recon	**98**	56	75	75	97
Game	**97**	64	82	86	96
Traffic	**99**	85	90	87	98
Crossing	88	58	78	83	**89**
Skill	**100**	25	71	69	86
Navigation	82	26	25	28	**83**
Total	**86**	56	70	72	83

Summary

Result Summary

- Hand made functions performed very well
- Default metareasoner severely underestimates thinking
- The improvements proved to be very useful

Summary

Outlook

- More general hand made functions
- Improve formal procedure:
  - Consider all previous ΔQ-values
  - Replace random ρ
- More sophisticated cost of thinking: combination of two approaches

Questions?

BRTDP vs UCT

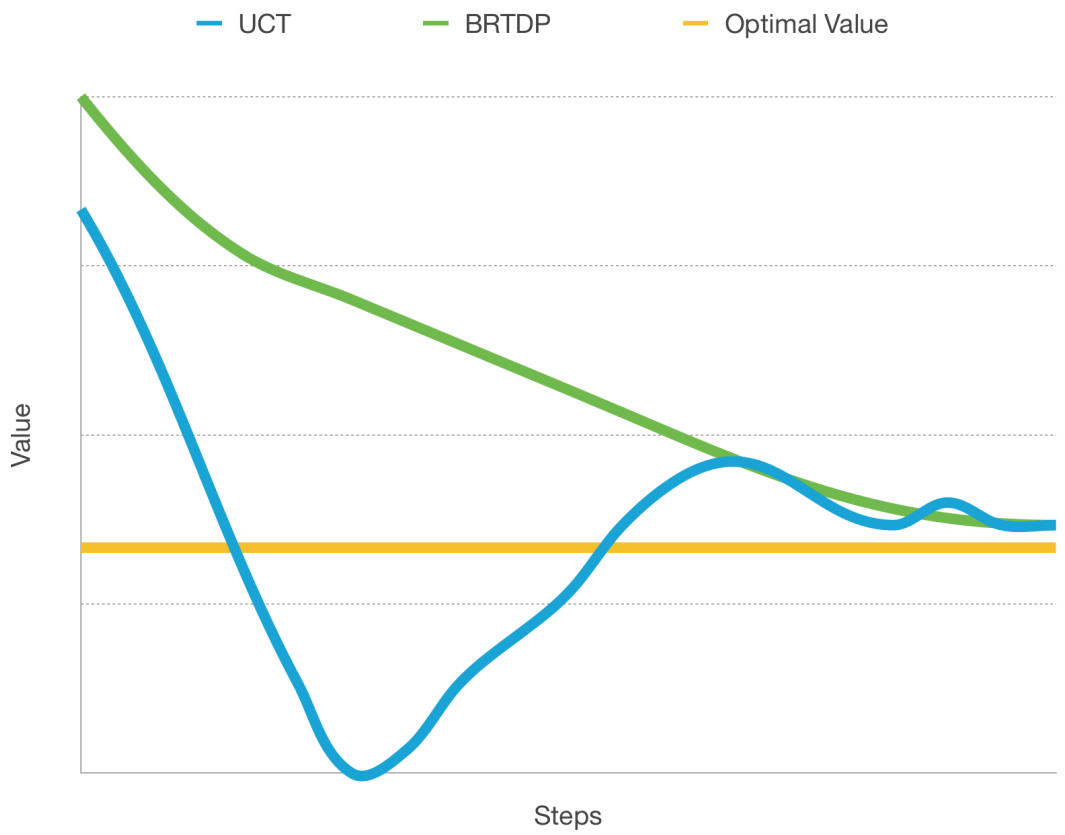
BRTDP

- Used in original paper
- Cost setting
- Uses upper bound of the actual Q-value
- Monotonously decreasing

UCT

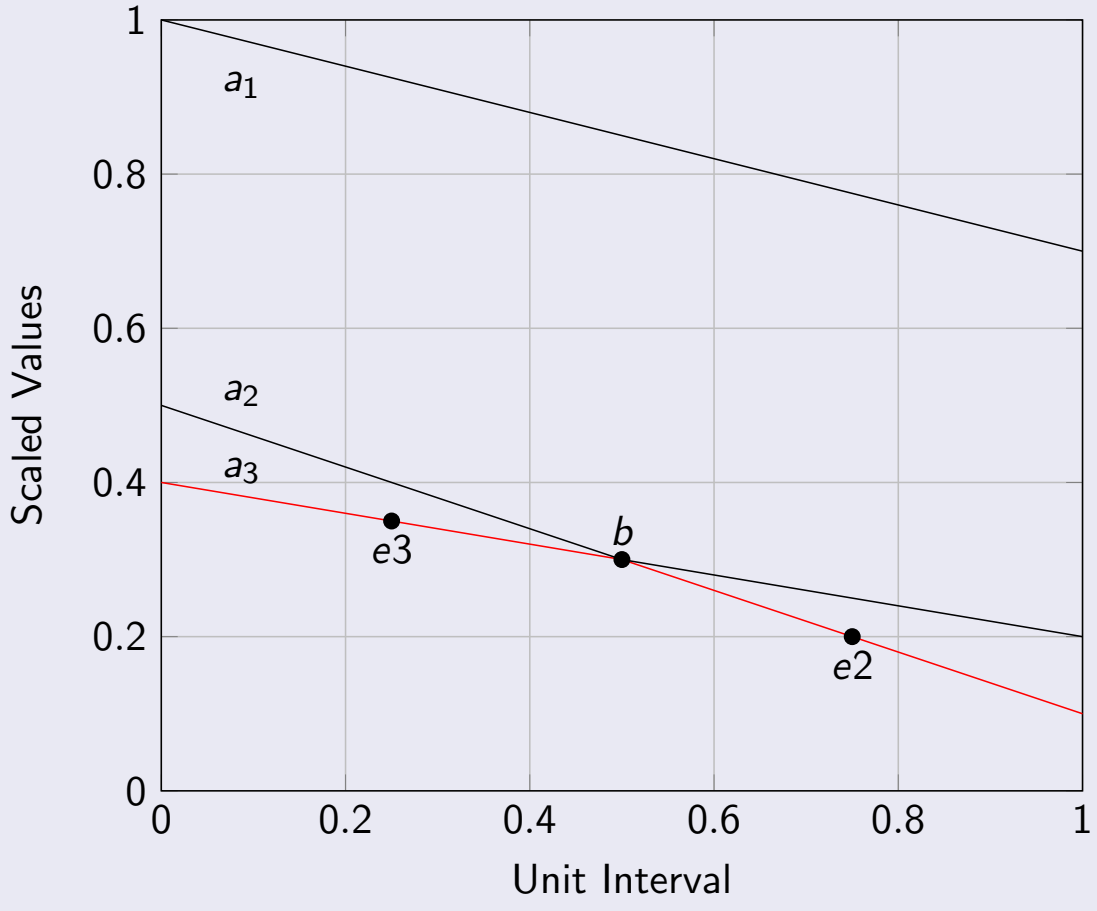
- Used by PROST planner
- Reward setting
- No guarantees

BRTDP vs UCT: Visualisation



Line Segment Example: BRTDP

$Q^{think} < Q^{act}$



Wildfire Time per Step