

# Monte Carlo Tree Search

**Branislav Božanský**

**PAH/PUI 2016/2017**

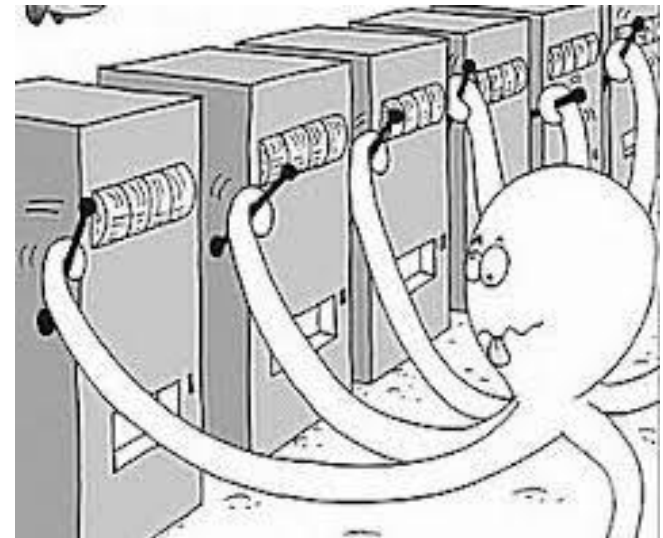
# MDPs – Using Monte Carlo Methods

---

- **Monte Carlo Simulation:** a technique that can be used to solve a mathematical or statistical problem using repeated sampling to determine the properties of some phenomenon (or behavior)
- **Monte-Carlo Planning:** compute a good policy for an MDP by interacting with an MDP simulator
- when simulator of a planning domain is available or can be learned from data
  - even if not described as a MDP
  - queries has to be cheap (relatively)

# MDPs – Using Monte Carlo Methods

- sequential decision problem (over a single state)
- $k \geq 2$  stochastic actions (arms  $a_i$ )
  - each parameterized with an unknown probability distribution  $v_i$
  - each with a stored expectation  $\mu_i$
  - if executed (pulled) rewarded at random from  $v_i$
- objective
  - get maximal reward after N pulls
  - minimize **regret** for pulling wrong arm(s)



- UCBI arm selection:
  - select arm  $a_i$  maximizing UCBI formula:

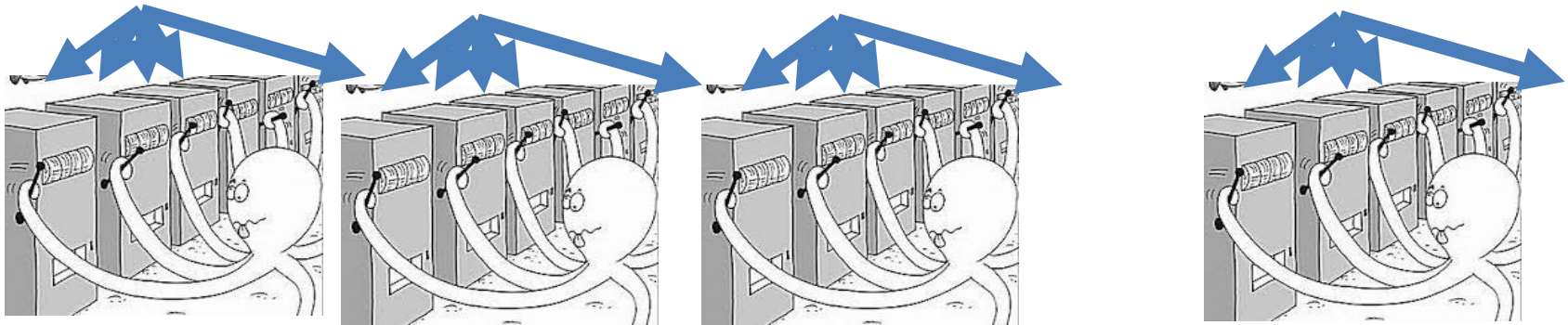
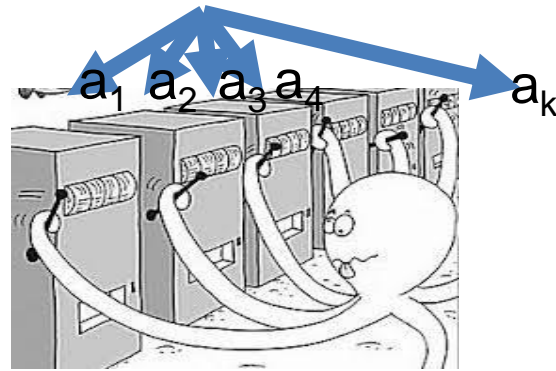
$$\mu_i + c \sqrt{\frac{\ln n}{n_i}}$$

and update  $\mu_i$

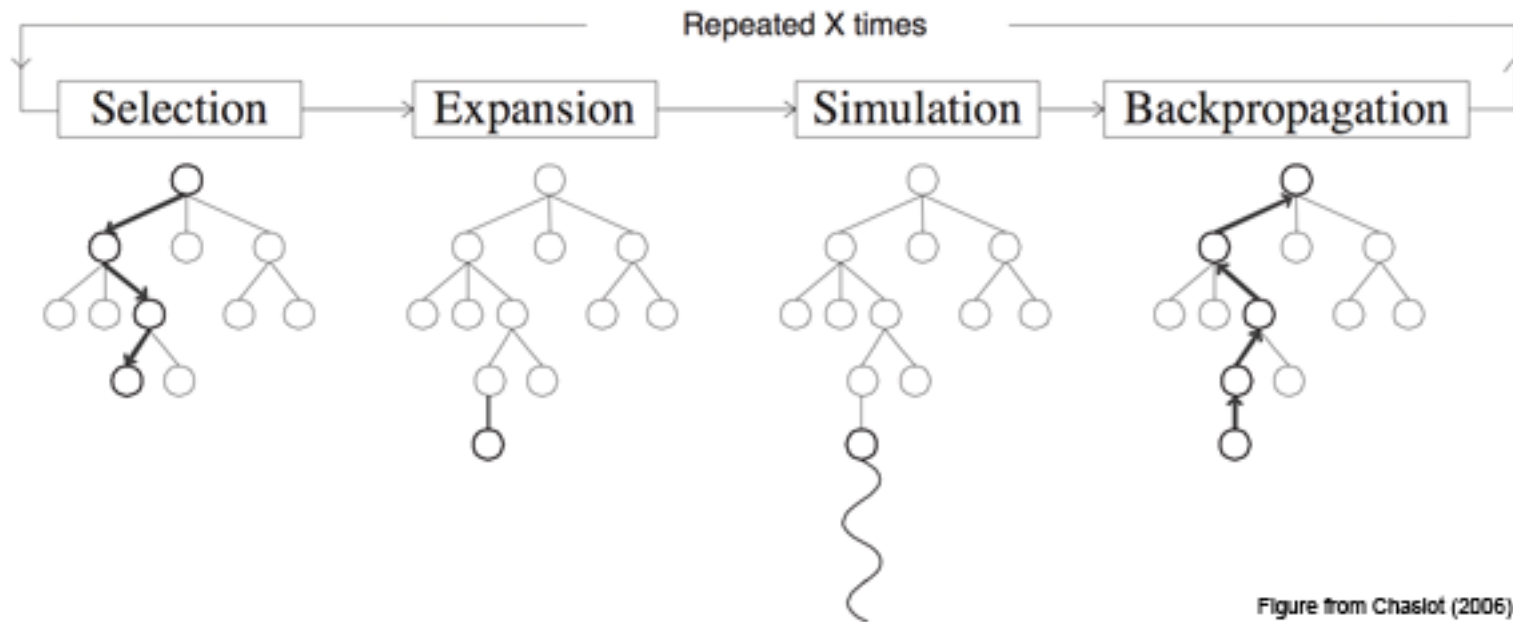
- $n$  – times the state is visited;  $n_i$  – times the action is visited
- $\mu_i$  – average reward from the previous plays
- exploration factor  $c$  ensures to evaluate actions that are evaluated rarely

# MCTS – from UCBI to UCT

- UCBI applied on trees – UCT

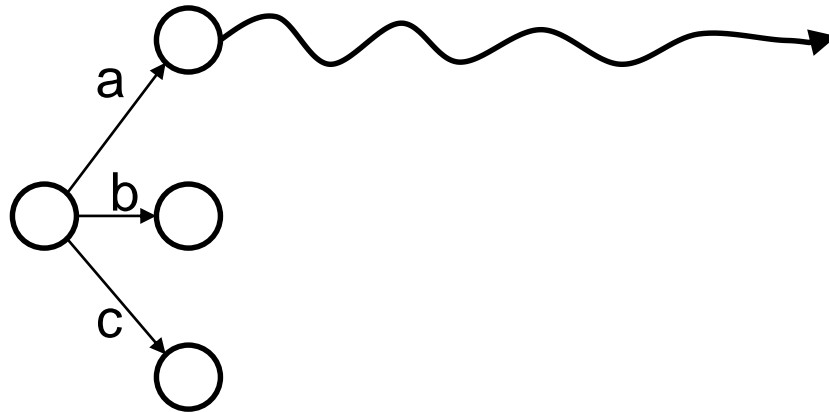


# MCTS – from UCBI to UCT



## Example

---



Random simulation results:

1. 2
2. 3
3. 1
4. 1
5. ...

**Questions (assume lexicographic tie breaking):**

1. Which action will be selected in the first iteration? **a**
2. Which action will be selected in the fourth iteration? **b**
3. Which action will be selected in the fifth iteration? **a**

# MCTS – PROST



IPPC 2011 winner

---

- Vanilla UCT does not work very well in practice
  - huge branching factor
  - long (infinite) horizon
  - very difficult to find the correct plan by random rollouts
- these issues were addressed by PROST
  - search depth limitation
  - pruning out unreasonable actions
  - heuristic value initialization



# MCTS – PROST



IPPC 2011 winner

---

- PROST – search depth limitation
  - we can limit search depth to  $L$  instead of solving to full depth
  - we need to do that if we have an infinite horizon
  - there can be a problem in re-using statistics from previous searches with limited depth (an optimal plan for horizon  $L$  does not have to be optimal for the full problem)
- PROST - pruning out unreasonable actions
  - we can heuristically identify unnecessary actions that do not yield any positive reward
  - compare to a NOOP action

# MCTS – PROST



IPPC 2011 winner

---

- PROST – initialization of values
  - vanilla UCT first evaluates an action, if this action has not been evaluated before in state  $s$
  - in case of a large branching factor, our search tree is very shallow
  - we can set some heuristic values to actions/children
  - we can set an artificial number of iterations
- we can set the values using some relaxation/determinization of the problem
  - Q-value initialization based on most probable outcome
  - the algorithm performs an iterative deepening search and checks whether the values are **informative** ( $I(s, a) > I(s, a_\emptyset)$ )

# MCTS – PROST



IPPC 2011 winner

---

	CROSSING	ELEVATORS	GAME	NAVIGATION	RECON	SKILL	SYSADMIN	TRAFFIC	TOTAL
$P^0$	0.46	0.01	0.86	0.14	0.00	0.89	0.86	0.98	$0.53 \pm 0.09$
$P$	0.51	0.04	0.91	0.27	0.40	0.90	0.86	0.96	$0.61 \pm 0.08$
$P_{15}$	0.56	0.01	<b>0.95</b>	0.30	0.46	0.91	<b>0.91</b>	<b>0.99</b>	$0.63 \pm 0.08$
$P^I$	0.84	0.86	0.88	0.65	<b>0.98</b>	0.94	0.82	0.84	$0.85 \pm 0.05$
$P_{15}^I$	0.83	0.93	0.91	0.57	<b>0.98</b>	<b>0.95</b>	0.88	0.93	$0.87 \pm 0.05$
$P^{I,R}$	<b>0.98</b>	0.85	0.86	0.71	<b>0.98</b>	0.89	0.80	0.83	$0.86 \pm 0.04$
$P_{15}^{I,R}$	0.91	<b>0.94</b>	0.92	0.67	0.97	0.92	0.86	0.94	<b>0.89</b> $\pm 0.04$
<b>Glutton</b>	0.80	0.90	0.67	<b>0.97</b>	0.76	0.86	0.34	0.67	$0.75 \pm 0.06$

# MCTS – Online planning

---

- Anytime algorithm
- A typical use case for MCTS-like approach is online planning – i.e. selecting the best action in the current situation in a limited time
- This corresponds to a simple regret – we do not want to regret not selecting a different action in the current state
- However, UCBI optimizes the cumulative regret (selecting the best arm over all attempts)
- But these attempts are fictitious in our case!
- While MAB approach works in practice, it does not exactly correspond to the online planning

- 
- There are two conflicting tasks
    - selecting the best action in state  $s$  (reaching  $s'$ )
    - exploring and finding the best continuation after  $s'$  is reached
  - In order to satisfy Task 2 – we need to select the best action sufficiently often
  - To do that, we need to know the optimal continuation
  - BRUE algorithm therefore two different action selection methods
    - the action in the selection phase is selected uniformly
    - the action in the update phase is selected using greedy strategy

```
procedure UPDATE( $\rho$ )
  for  $d \leftarrow |\rho|, \dots, 1$  do
     $h \leftarrow H - d$ 
     $\langle s, a, r, s' \rangle \leftarrow \rho[d]$ 
     $n(s\langle h \rangle) \leftarrow n(s\langle h \rangle) + 1$ 
     $n(s\langle h \rangle, a) \leftarrow n(s\langle h \rangle, a) + 1$ 
     $n(s\langle h \rangle, a, s') \leftarrow n(s\langle h \rangle, a, s') + 1$ 
     $\bar{r} \leftarrow r + \text{ESTIMATE}(s'\langle h - 1 \rangle)$ 
    MC-BACKUP( $s\langle h \rangle, a, \bar{r}$ )
```

```
procedure ESTIMATE( $s\langle h \rangle$ )
   $\bar{r} \leftarrow 0$ 
  for  $d \leftarrow 0, \dots, h - 1$  do
     $a \leftarrow \text{ESTACTION}(s\langle h - d \rangle)$ 
     $s' \leftarrow \text{ESTOUTCOME}(s\langle h - d \rangle, a)$ 
     $\tilde{r}_{d+1} \leftarrow R(s, a, s')$ 
     $\bar{r} \leftarrow \bar{r} + \tilde{r}_{d+1}$ 
     $s \leftarrow s'$ 
  return  $\bar{r}$ 
```

```
procedure STOPROLLOUT( $\rho$ )
   $d \leftarrow |\rho|$ 
  return  $d = H$  or  $A(\rho[d].s') = \emptyset$ 

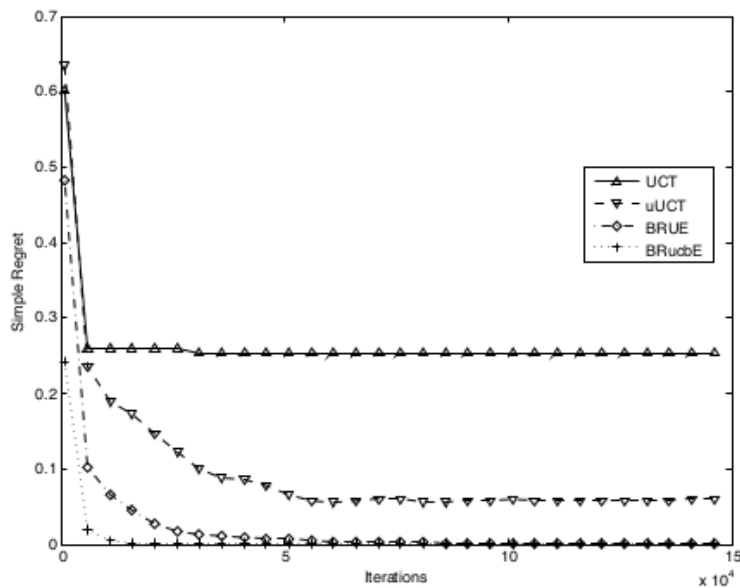
procedure ROLLOUTACTION( $s\langle h \rangle$ ) // uniform
  return  $a \sim \mathcal{U}[A(s)]$ 

procedure ROLLOUTOUTCOME( $s\langle h \rangle, a$ )
  return  $s' \sim \mathcal{P}(S | s, a)$ 

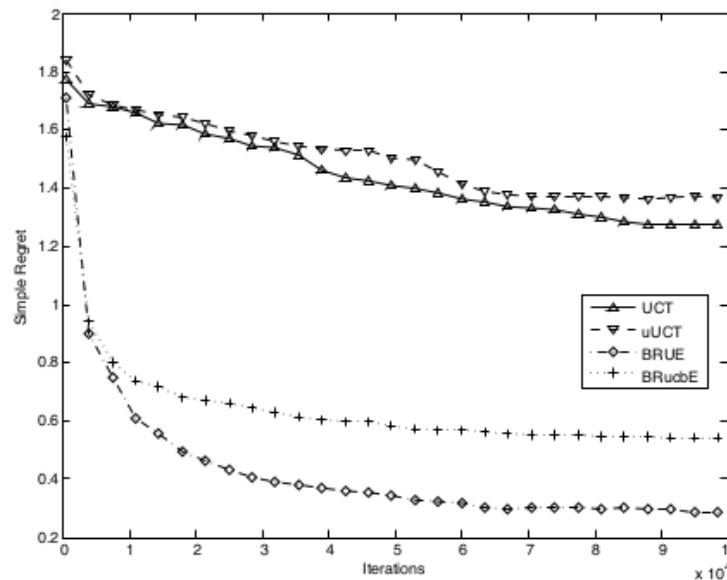
procedure ESTACTION( $s\langle h \rangle$ ) // best
  return  $\operatorname{argmax}_{a \in A(s)} \hat{Q}(s\langle h \rangle, a)$ 

procedure ESTOUTCOME( $s\langle h \rangle, a$ )
  for  $s' : n(s\langle h \rangle, a, s') > 0$  do
     $\hat{\mathcal{P}}(S = s' | s, a) \leftarrow \frac{n(s\langle h \rangle, a, s')}{n(s\langle h \rangle, a)}$ 
  return  $s' \sim \hat{\mathcal{P}}(S | s, a)$ 
```

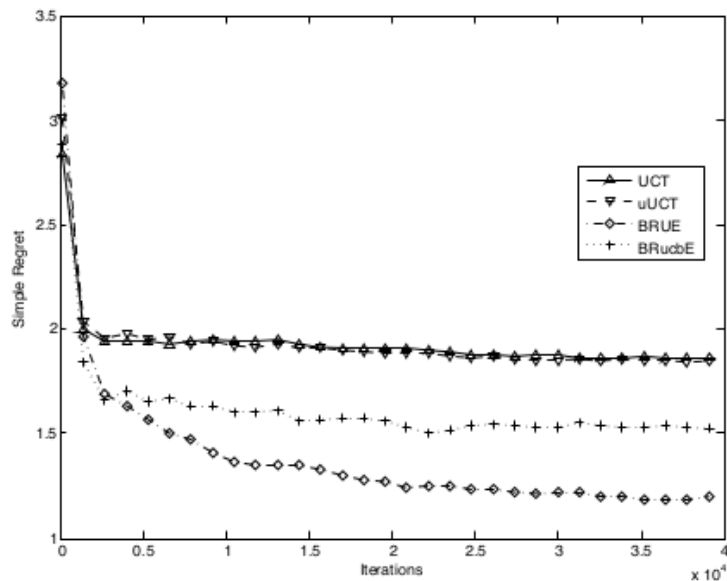
# MCTS – BRUE



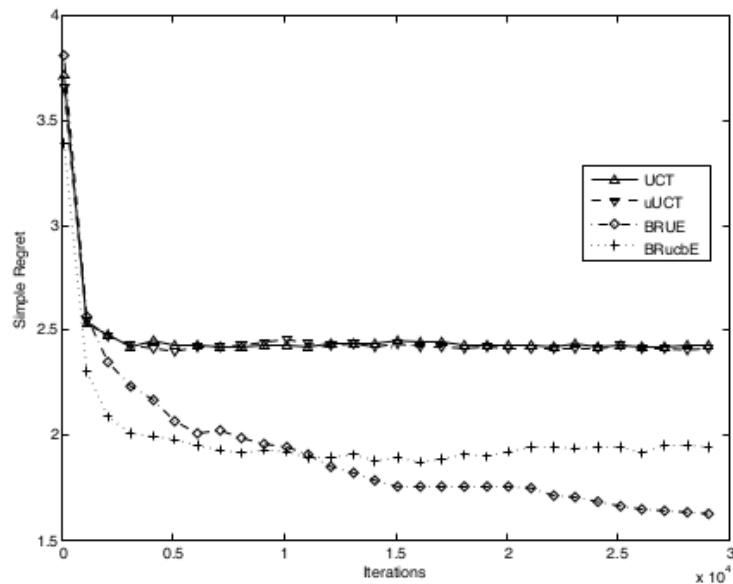
5 × 5



10 × 10



20 × 20



40 × 40

# Trial-based Heuristic Tree Search (THTS)

---



- a common framework based on five ingredients:
  - heuristic function
  - backup function
  - action selection
  - outcome selection
  - trial length
- subsuming: MCTS, UCT, FIND-and-REVISE, AO\* (AND/OR graph solver), Real-Time Dynamic Programming (RTDP), various heuristic functions (e.g., iterative deepening search)
- providing: MaxUCT, UCT\*, ...
- UCT\* in PROST 2014 is currently best performing IPPC planner



# Trial-based Heuristic Tree Search (THTS)



---

**Algorithm 1:** The THTS schema.

---

```
1 THTS(MDP  $M$ , timeout  $T$ ):
2    $n_0 \leftarrow \text{getRootNode}(M)$ 
3   while not solved( $n_0$ ) and time() <  $T$  do
4     visitDecisionNode( $n_0$ )
5   return greedyAction( $n_0$ )

6 visitDecisionNode(Node  $n_d$ ):
7   if  $n_d$  was never visited then initializeNode( $n_d$ )
8    $N \leftarrow \text{selectAction}(n_d)$ 
9   for  $n_c \in N$  do
10    visitChanceNode( $n_c$ )
11  backupDecisionNode( $n_d$ )

12 visitChanceNode(Node  $n_c$ ):
13   $N \leftarrow \text{selectOutcome}(n_c)$ 
14  for  $n_d \in N$  do
15    visitDecisionNode( $n_d$ )
16  backupChanceNode( $n_c$ )
```

---

# Trial-based Heuristic Tree Search (THTS)



- maintains explicit tree of alternating decision and chance nodes
- selection phase
  - alternating **visitDecisionNode** and **visitChanceNode**
  - selection by **selectAction** and **selectOutcome**
  - tree traversing (down)
- expansion phase
  - when unvisited node encountered
  - add child node for each action
  - heuristics used to initialize the estimates
  - allows selection phase for new nodes

---

**Algorithm 1:** The THTS schema.

---

```
1 THTS(MDP  $M$ , timeout  $T$ ):
2    $n_0 \leftarrow \text{getRootNode}(M)$ 
3   while not solved( $n_0$ ) and time() <  $T$  do
4     visitDecisionNode( $n_0$ )
5   return greedyAction( $n_0$ )
6 visitDecisionNode(Node  $n_d$ ):
7   if  $n_d$  was never visited then initializeNode( $n_d$ )
8    $N \leftarrow \text{selectAction}(n_d)$ 
9   for  $n_c \in N$  do
10    visitChanceNode( $n_c$ )
11    backupDecisionNode( $n_d$ )
12 visitChanceNode(Node  $n_c$ ):
13    $N \leftarrow \text{selectOutcome}(n_c)$ 
14   for  $n_d \in N$  do
15     visitDecisionNode( $n_d$ )
16   backupChanceNode( $n_c$ )
```

---

# Trial-based Heuristic Tree Search (THTS)



- selection and expansion phases alternate until the **trial length**
- backup phase (**backupDecisionNode** & **backupChanceNode**)
  - all selected nodes are updated in reverse order
  - when another selected, but not yet visited  $\rightarrow$  selection phase
  - a trial ends when the backup is called on the root node
  - tree backing (up)
- the process is repeated until the timeout  $T$  allows for another trial
- highest expectation action is returned **greedyAction**

---

**Algorithm 1:** The THTS schema.

---

```
1 THTS(MDP  $M$ , timeout  $T$ ):
2    $n_0 \leftarrow \text{getRootNode}(M)$ 
3   while not solved( $n_0$ ) and time() <  $T$  do
4     visitDecisionNode( $n_0$ )
5   return greedyAction( $n_0$ )
6 visitDecisionNode(Node  $n_d$ ):
7   if  $n_d$  was never visited then initializeNode( $n_d$ )
8    $N \leftarrow \text{selectAction}(n_d)$ 
9   for  $n_c \in N$  do
10    visitChanceNode( $n_c$ )
11    backupDecisionNode( $n_d$ )
12 visitChanceNode(Node  $n_c$ ):
13    $N \leftarrow \text{selectOutcome}(n_c)$ 
14   for  $n_d \in N$  do
15     visitDecisionNode( $n_d$ )
16   backupChanceNode( $n_c$ )
```

---

# Trial-based Heuristic Tree Search (THTS)



- 
- Heuristic function
    - action value initialization (Q-value)
$$h: S \times A \mapsto \mathbb{R}$$
    - state value initialization (V-value)
$$h: S \mapsto \mathbb{R}$$
  - Action selection
    - UCBI,  $\epsilon$ -greedy, ...
  - Outcome selection
    - Monte Carlo sampling; outcome based on biggest potential impact

# Trial-based Heuristic Tree Search (THTS)



## Backup

---

- optimal policy derived from the **Bellman optimality equation**:

$$V^*(s) = \begin{cases} 0 & \text{if } s \text{ is terminal} \\ \max_{a \in A} Q^*(a, s) & \text{otherwise} \end{cases}$$

$$Q^*(a, s) = R(a, s) + \sum_{s' \in S} P(s' | a, s) \cdot V^*(s')$$

- **Full Bellman backup** ~ Bellman optimality equation,  $k$  trials
- **Monte Carlo backup**

$$V^k(s) = \begin{cases} 0 & \text{if } s \text{ is terminal} \\ \frac{\sum_{a \in A} n_{a,s} \cdot Q^k(a, s)}{n_s} & \text{otherwise} \end{cases}$$

$$Q^k(a, s) = R(a, s) + \frac{\sum_{s' \in S} n_{s'} \cdot V^k(s')}{n_{a,s}}$$

- 
- backup function
    - action-value by Monte Carlo backup ( $Q^k(s)$ )
    - state-value by Full Bellman backup ( $V^*(s)$ )
  - action selection → UCBI
  - outcome selection → Monte Carlo sampling (MDP based)
  - heuristic function → N/A
  - trial length → UCT (horizon length, i.e. to leafs)

- 
- backup function
    - Partial Bellman backup  
(weighted proportionally to subtree probability)
  - action selection → UCBI
  - outcome selection → Monte Carlo sampling (MDP based)
  - heuristic function → Iterative Deepening Search (depth: 15)
  - trial length → explicit tree length + 1  
(only initialized new nodes using heuristics)
  - resembles classical heuristic Breadth-First-Search (rather than UCT Depth-First-Search)

	ELEVATORS	SYSADMIN	RECON	GAME	TRAFFIC	CROSSING	SKILL	NAVIGATION	Total
<b>UCT</b>	0.93	0.66	<b>0.99</b>	0.88	0.84	0.85	0.93	0.81	0.86
<b>MaxUCT</b>	<b>0.97</b>	0.71	0.88	0.9	0.86	<b>0.96</b>	0.95	0.66	0.86
<b>DP-UCT</b>	<b>0.97</b>	0.65	0.89	0.89	0.87	<b>0.96</b>	<b>0.98</b>	<b>0.98</b>	0.9
<b>UCT*</b>	<b>0.97</b>	<b>1.0</b>	0.88	<b>0.98</b>	<b>0.99</b>	<b>0.98</b>	<b>0.97</b>	<b>0.96</b>	<b>0.97</b>
<b>PROST</b>	0.93	0.82	<b>0.99</b>	0.93	0.93	0.82	<b>0.97</b>	0.55	0.87



# References

---

- Keller & Eyerich “PROST: Probabilistic Planning Based on UCT” ICAPS 2012
- Feldman & Domshlak “Simple Regret Optimization in Online Planning for Markov Decision Processes” JAIR 2014
- Keller & Helmert “Trial-based Heuristic Tree Search for Finite Horizon MDPs” ICAPS 2013