Online Saturated Cost Partitioning for Classical Planning

Jendrik Seipp
August 2021
Linköping University, University of Basel
• optimal classical planning
• $A^*$ search + admissible heuristic
• multiple abstraction heuristics
• cost partitioning
• optimal classical planning
• $A^*$ search + admissible heuristic
• multiple abstraction heuristics
• saturated cost partitioning
different states need different cost partitionings:

• precompute cost partitionings
  → no good stopping criterion, search starts late

• compute cost partitioning for each state
  → too expensive
Coverage over time

- offline-1000s
- online-each-state
Coverage over time

- offline-1000s
- online-each-state
- online-1000s
Cost partitioning

- **split action costs** among heuristics such that: sum of costs $\leq$ original cost

Saturated cost partitioning

- order heuristics, then for each heuristic $h$:
  - use **minimum costs** preserving all estimates of $h$
  - use **remaining costs** for subsequent heuristics
Cost partitioning

- **split action costs** among heuristics such that: sum of costs \( \leq \) original cost

Saturated cost partitioning

- order heuristics, then for each heuristic \( h \):
  - use **minimum costs** preserving all estimates of \( h \)
  - use **remaining costs** for subsequent heuristics
Cost partitioning

- **split action costs** among heuristics such that: sum of costs $\leq$ original cost

Saturated cost partitioning

- order heuristics, then for each heuristic $h$:
  - use **minimum costs** preserving all estimates of $h$
  - use **remaining costs** for subsequent heuristics
**Background**

**Cost partitioning**
- **split action costs** among heuristics such that: sum of costs ≤ original cost

**Saturated cost partitioning**
- order heuristics, then for each heuristic $h$:
  - use **minimum costs** preserving all estimates of $h$
  - use **remaining costs** for subsequent heuristics
Background

Cost partitioning
- **split action costs** among heuristics such that: sum of costs $\leq$ original cost

Saturated cost partitioning
- order heuristics, then for each heuristic $h$:
  - use **minimum costs** preserving all estimates of $h$
  - use **remaining costs** for subsequent heuristics

![Diagram showing cost partitioning with states $S_1, S_2, S_3, S_4, S_5$ and costs 0, 1, 0, 0, 3, 4, 1, 0, 0, 0, 0, 0, 0]
Order matters:

- $h^{SCP}(s_2) = 8$
- $h^{SCP}(s_2) = 7$
Order matters:

- $h^{SCP}(s_2) = 8$
- $h^{SCP}(s_2) = 7$

→ use multiple orders and maximize over estimates
Offline diversification

- sample 1000 states
- start with empty set of orders
- until time limit is reached:
  - compute order for new sample
  - store order if a sample profits from it
Online diversification

**COMPUTEHEURISTIC(s)**

- if SELECT(s) and not time limit reached
  - compute order for s
  - store order if s profits from it
- return maximum over all stored orders for s
Offline vs. online diversification

Offline
• compute orders for samples for \( T \) seconds
• store order if one of 1000 samples profits from it

Online
• compute orders for subset of evaluated states for at most \( T \) seconds
• store order if single evaluated state profits from it
Selection strategies

**Select**

- Bellman (Eifler and Fickert 2018)
- Novelty (Lipovetzky and Geffner 2012)
- Interval
Coverage over time

- offline-1000s
- online-each-state
- online-1000s

solved tasks vs. time in seconds

- 10^0
- 10^1
- 10^2
- 10^3
<table>
<thead>
<tr>
<th>Offline diversification</th>
<th>Online computation</th>
<th>Online diversification</th>
</tr>
</thead>
<tbody>
<tr>
<td>long precomputation</td>
<td>no precomputation</td>
<td>no precomputation</td>
</tr>
<tr>
<td>samples</td>
<td>states</td>
<td>states</td>
</tr>
<tr>
<td>fast evaluations</td>
<td>slow evaluations</td>
<td>fast evaluations</td>
</tr>
<tr>
<td>high coverage</td>
<td>low coverage</td>
<td>high coverage</td>
</tr>
</tbody>
</table>