Improved Pattern Selection for PDB Heuristics in Classical Planning (Extended Abstract)

Sascha Scherrer  Florian Pommerening  Martin Wehrle
University of Basel, Switzerland

iPDB (Haslum et al. 2007)
- State-of-the-art pattern selection algorithm
- Selects patterns (sets of variables) using hill-climbing search in the space of pattern collections
- Canonical heuristic of resulting pattern collection used in A*-search

Local Maxima in iPDB
- Hill-climbing can terminate early in local maximum
- No extension with one variable has sufficient improvement
- Well-known but unaddressed problem (already pointed out by Haslum et al. 2007)

iPDB with Variable Neighborhood Search
- Addresses the problem of local maxima
- Based on variable neighborhood search (Mladenovic and Hansen 1997)
- Looks for successor collections of increasing size
- Extends existing candidate patterns by further causally related variables
- Resets candidate collection once an improving candidate is found
- Anytime algorithm: runs as long as resources are available
- We limit resources to stop the hill-climbing

Experimental Evaluation
- Evaluated on IPC tasks
- optimal tracks 1998–2011
- Resource limits
  - Very important to limit both time and memory
  - Robust to parameter changes
- iPDB-VNS improves iPDB
- Heuristic quality
- Number of solved tasks
- iPDB-VNS is competitive with LM-cut

Heuristic Quality

| Problem  | #Unsolved | #Unsolved Optimal | M | M
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Airport (50)</td>
<td>25, 38, 38</td>
<td>28</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Depot (22)</td>
<td>8, 8, 11</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elevators (50)</td>
<td>36, 36, 43</td>
<td>40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Florist (20)</td>
<td>2, 2, 4</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Miconic (150)</td>
<td>55, 55, 55</td>
<td>141</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parchimter (50)</td>
<td>22, 28, 28</td>
<td>31</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TPP (30)</td>
<td>6, 6, 8</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transport (30)</td>
<td>17, 17, 24</td>
<td>17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trucks (30)</td>
<td>8, 8, 10</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Woodworking (60)</td>
<td>13, 23, 23</td>
<td>29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sum (502)</td>
<td>192, 221, 244</td>
<td>317</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sum in other domains (694)</td>
<td>474, 473, 481</td>
<td>452</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Total sum (1396) 666, 694, 725, 769
Coverage score (in %) 50.72 52.68 55.45 53.60

Pseudocode (iPDB)

```python
function generate-candidates(P)
    Candidates := φ
    for P ∈ P :
        for V ∈ causally-related(V) \ P :
            Candidates := Candidates ∪ {P ∪ {V}}
    return Candidates

function iPDB()
    P := {{V_i} | V_i is a goal variable}
    Candidates := generate-candidates(P)
    while True :
        S := generate-samples(1000)
        for P_c ∈ Candidates :
            improvement[P_c] := |{s ∈ S | h(P_c)|s) > h^*(s)}|
        P_best := Candidate with highest improvement
        if improvement[P_best] > threshold :
            P := P ∪ {P_best}
            Candidates := generate-candidates(P)
        else :
            return h^*
```

Pseudocode (iPDB-VNS)

```python
function generate-candidates(P)
    Candidates := φ
    for P ∈ P :
        for V ∈ causally-related(V) \ P :
            Candidates := Candidates ∪ {P ∪ {V}}
    return Candidates

function iPDB-VNS()
    P := {{V_i} | V_i is a goal variable}
    Candidates := generate-candidates(P)
    while True :
        S := generate-samples(1000)
        for P_c ∈ Candidates :
            improvement[P_c] := |{s ∈ S | h(P_c)|s) > h^*(s)}|
        P_best := Candidate with highest improvement
        if improvement[P_best] > threshold :
            P := P ∪ {P_best}
            Candidates := generate-candidates(P)
        else :
            Candidates := generate-candidates(Candidates)
            if time or memory limit exceeded :
                return h^*
```

#Expansions during A* Search

![Graph showing the number of expansions during A* search](image-url)