Title: Dual Viewpoint Heuristics for Binary Constraint Satisfaction Problems
Author: P.A. Geelen
Proc.: ECAI 1992
Pages: 31-35

Foundations of Constraint Satisfaction
CSCE421/821, Fall 2004
www.cse.unl.edu/~choueiry/F04-421-821/

Berthe Y. Choueiry (Shu-we-ri)
Avery Hall, Room 123B
choueiry@cse.unl.edu, Tel: (402)472-5444
Contributions

I- New heuristics for variable & value selection

II- Double-viewpoint strategy
    (common in scheduling: job vs. resource-centered perspective)
    [Sadeh ’91]

III- Validation on the $n$-Queen problem

Assumptions

- Binary constraints, finite domains
- Seeking **one** solution (relevant for value ordering)
- Using backtracking (BT) and forward-checking (FC)
I- Goal of Var/Val orderings in BT

- avoid constraint violation
  → select values that do not cause constraint violation
  → most promising value first

- discover constraint violation quickly
  → select variables that do not delay constraint violation
  → most constrained variable first (fail-first principle)
Past variables
\[ X_1 = V_1 \]
\[ X_2 = V_2 \]
\[ X_h = V_h \]

Future variables
\[
\begin{align*}
&X_i, \text{DOM}_S(X_i) \\
&X_j \\
&X_n
\end{align*}
\]

After forward-checking given \( S \) \[
\begin{cases}
\text{DOM}_S(X_i) \\
\vdots \\
\text{DOM}_S(X_j) \\
\text{DOM}_S(X_n)
\end{cases}
\]

For \( X_i = V_i \), we can separate \( \text{DOM}_S(X_j) \) into two sets in \( \mathcal{O}(a) \):

- values consistent with \( X_i = V_i \)
  \[ \rightarrow \text{ of size } \text{LEFT}_S(X_j \mid X_i = V_i) \]

- values inconsistent with \( X_i = V_i \)
  \[ \rightarrow \text{ of size } \text{LOST}_S(X_j \mid X_i = V_i) \]
Value selection for $X_i$

Choose the least constraining value $V_i$

1. Minimize $X_j$ future variables

$$\text{Cost}_S(X_i = V_i) = \sum_{X_j \neq i} \text{LOST}_S(X_j \mid X_i = V_i)$$

2. Minimize $X_j$ future variables

$$\text{Cruciality}_S(X_i = V_i) = \sum_{X_j \neq i} \frac{\text{LOST}_S(X_j \mid X_i = V_i)}{|\text{DOM}_S(X_j)|}$$

3. Maximize $X_j$ future variables

$$\text{Promise}_S(X_i = V_i) = \prod_{X_j \neq i} \text{LEFT}_S(X_j \mid X_i = V_i)$$

$\rightarrow$ number of assignments that $X_i = V_i$ and can be done such that no constraint on $X_i$ is violated

(3) is more discriminating ($\text{LEFT}_S(X_j \mid X_i = V_i) = 0$)
Value selection: example

- Minimize \(X_j\) future variables

\[
\text{Cost}_S(X_i = V_i) = \sum_{X_j \neq i} \text{LOST}_S(X_j \mid X_i = V_i)
\]

| X1 | 6 | 6 | 6 | 6 |
| X2 | 6 | 8 | 8 | 6 |
| X3 | 6 | 8 | 8 | 6 |
| X4 | 6 | 6 | 6 | 6 |

- Maximize \(X_j\) future variables

\[
\text{Promise}_S(X_i = V_i) = \prod_{X_j \neq i} \text{LEFT}_S(X_j \mid X_i = V_i)
\]

| X1 | 8 | 6 | 6 | 8 |
| X2 | 8 | 2 | 2 | 8 |
| X3 | 8 | 2 | 2 | 8 |
| X4 | 8 | 6 | 6 | 8 |
Variable selection

Choose the most constrained variable (FFP) $X_i$

1. Least domain (LD)

2. Maximize $V_i \in \text{DOM}_S(X_i)$
   \[
   \text{Criticality}_S(X_i) = \frac{1}{\prod_{V_i} (1+|\text{DOM}_S(X_i)| \times \text{Cruciality}_S(X_i = V_i))}
   \]

3. Minimize $V_i \in \text{DOM}_S(X_i)$
   \[
   \text{Promise}_S(X_i) = \sum_{V_i} \text{Promise}_S(X_i = V_i)
   \]

→ number of assignments that can be done such that no constraint on $X_i$ is violated
Variable selection: example

Minimize:

\[ V_i \in \text{DOM}_S(X_i) \]

\[ \text{Promise}_S(X_i) = \sum_{V_i} \text{Promise}_S(X_i = V_i) \]

<table>
<thead>
<tr>
<th></th>
<th>8</th>
<th>6</th>
<th>6</th>
<th>8</th>
<th>28</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X2</td>
<td>8</td>
<td>2</td>
<td>2</td>
<td>8</td>
<td>20</td>
</tr>
<tr>
<td>X3</td>
<td>8</td>
<td>2</td>
<td>2</td>
<td>8</td>
<td>20</td>
</tr>
<tr>
<td>X4</td>
<td>8</td>
<td>6</td>
<td>6</td>
<td>8</td>
<td>28</td>
</tr>
</tbody>
</table>

\( X_1, X_4 \) promise 28 solutions
\( X_2, X_3 \) promise 20 solutions, more constraining

Start with \( X_2 \) or \( X_3 \) (more constraining) and, choose columns 1 or 4 (more promising)
Summary

Most promising value:

(1) Minimum cost
(2) Minimum cruciality
(3) Maximum promise

Most constrained variable:

- Least domain (LD)
(4) Maximum criticality
(5) Minimum (false) promise

→ Dynamic variable/value orderings
Algorithms

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Choice of Var</th>
<th>Choice of val</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD+1</td>
<td>least domain</td>
<td>Minimum cost</td>
</tr>
<tr>
<td>LD+2</td>
<td>–</td>
<td>Min. cruciality</td>
</tr>
<tr>
<td>LD+3</td>
<td>–</td>
<td>Max promise</td>
</tr>
<tr>
<td>FE+2/4</td>
<td>Max. critical</td>
<td>Min. cruciality</td>
</tr>
<tr>
<td>FE+3/5</td>
<td>Min. promise</td>
<td>Max. promise</td>
</tr>
</tbody>
</table>

LD: time $\mathcal{O}(n^2a^2)$, space $\mathcal{O}(na)$
FE: time $\mathcal{O}(n^2a^2)$, space $\mathcal{O}(na)$

Implementation hack: domino effect, saves computations.
II- Permutation problems

\[ n = a \]

\[ \rightarrow \text{constraint graph is complete} \]
\[ \text{constraints are MUTEX, All-diffs} \]

\[ \rightarrow \text{matching, marriage problem} \]
\[ \text{find one value for every variable and} \]
\[ \text{exactly one variable for every value} \]

Viewpoints: variable vs. values

\[ \rightarrow \text{Which viewpoint to take?} \]
\[ \rightarrow \text{How to combine computations in viewpoints?} \]
Permutation problems (cont’d)

At any point in BT,

\[ \# \text{ future variables} = \# \text{ future values} \]

→ choose the most constrained variable in the var-viewpoint, except when the most constrained value in the val-viewpoint is more constrained

PP: Least domain

<table>
<thead>
<tr>
<th></th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
<th>X4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>X1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
PP: Full evaluation

Introduce: \[ \left\{ \begin{array}{l} \text{LEFT}^{inv}(V_i|X_i = V_i) \\
\text{Promise}^{inv}(V_i = X_i) \end{array} \right. \]

Combine viewpoints: \( \text{CPromise}(X_i = V_i) \) min. \[ \left\{ \begin{array}{l} \text{Promise}(X_i = V_i) \\
\text{Promise}^{inv}(V_i = X_i) \end{array} \right. \]

Evaluate vars/vals using: \[ \left\{ \begin{array}{l} \text{CPromise}(X_i) = \sum_k \text{CPromise}(X_i = V_k) \\
\text{CPromise}^{inv}(V_i) = \sum_k \text{CPromise}(X_k = V_i) \end{array} \right. \]

Partial permutation problems \((n \leq a)\)

\(\rightarrow\) fake a real PP with bogus variables

\(\rightarrow\) extend (3) for PPP
III- Experiments

- 100 $N$-queen problems: $4 \leq N \leq 103$

- Comparison criteria
  - average number of backtrack
  - number of backtrack-free solutions
  - maximum number of backtracks
  - number of constraint checks

Algorithms: 
\[
\begin{aligned}
LD-1, LD-2, LD-3, LD-1+Dual \\
FE-2-4, FE-3-5, FE-3-5+Dual \\
FP-2-4, FP-3-5
\end{aligned}
\]
<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Average</th>
<th>#backtrack-free solutions</th>
<th>Max.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>#backtrack</td>
<td></td>
<td>#bactrack</td>
</tr>
<tr>
<td>LD+formula 1</td>
<td>&gt;45000</td>
<td>20</td>
<td>&gt;250000</td>
</tr>
<tr>
<td>LD+formula 2</td>
<td>&gt;33000</td>
<td>15</td>
<td>&gt;2500000</td>
</tr>
<tr>
<td>LD+formula 3</td>
<td>&gt;1205</td>
<td>3</td>
<td>92379</td>
</tr>
<tr>
<td>LD+formula 1, dual</td>
<td>21.6</td>
<td>26</td>
<td>548</td>
</tr>
<tr>
<td>FE+formulae 2&amp;4</td>
<td>9.5</td>
<td>71</td>
<td>812</td>
</tr>
<tr>
<td>FE+formulae 3&amp;5</td>
<td>5.1</td>
<td>68</td>
<td>266</td>
</tr>
<tr>
<td>FP+formulae 2&amp;4</td>
<td>5.6</td>
<td>81</td>
<td>496</td>
</tr>
<tr>
<td>FP+formulae 3&amp;5</td>
<td>4.2</td>
<td>68</td>
<td>224</td>
</tr>
<tr>
<td>FE+formulae 3&amp;5, dual</td>
<td>0.38</td>
<td>90</td>
<td>12</td>
</tr>
</tbody>
</table>

Dual perspective exhibits dramatic improvements

CPU time and #CC: plot hard to read, not interpreted:
→ Number of #CC is prohibitive in practice
→ Computations do FC implicitly.
    Can be exploited by bookkeeping.