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 2005
`www.cse.unl.edu/~choueiry/F05-421-821/`

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]

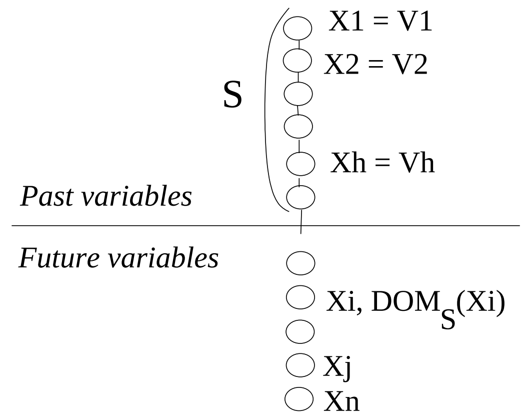
III- Validation on the n-Queen problem

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)



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 $X_i = V_i$
  $\rightarrow$ of size $\text{LEFT}_S(X_j \mid X_i = V_i)$

- values inconsistent with $X_i = V_i$
  $\rightarrow$ 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 $\qquad X_j$ 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 $\qquad X_j$ 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 $\qquad X_j$ 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 ($\text{LEFT}_S(X_j \mid X_i = V_i) = 0$)

Advantages of Promise for value selection

1. Product recognizes domain wipe-out, sum does not.
   Compare: 6+0 and 6×0
2. Product discriminates better than sum.
   6+0, 5+1, 4+2, 3+3 are all equivalent.
   However, (6×0) < (5×1) < (4×2) < (3×3)
   (the product of two numbers is larger as their average is larger)
3. Promise has a 'semantic' (i.e., physical interpretation)
   Upper bound on number of solutions.

These advantages are unique to Promise.

Value selection: example

- Minimize $\qquad X_j$ 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 $\qquad X_j$ 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 $\qquad V_i \in \text{DOM}_S(X_i)$

   $\text{Criticality}_S(X_i) =$

   $$\prod_{V_i} \frac{1}{(1+|\text{DOM}_S(X_i)| \times \text{Cruciality}_S(X_i = Vi))}$$

3. Minimize $\qquad V_i \in \text{DOM}_S(X_i)$

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

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

Variable selection: example

Minimize: $V_i \in \mathrm{DOM}_S(X_i)$

$$\mathrm{Promise}_S(X_i) = \sum_{V_i} \mathrm{Promise}_S(X_i = V_i)$$

X1	8	6	6	8	28
X2	8	2	2	8	20
X3	8	2	2	8	20
X4	8	6	6	8	28

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 *[Keng & Yun, 89]*
- (3) Maximum promise *[Geelen'92]*

Most constrained variable:
- • Least domain (LD)
- (4) Maximum criticality *[Keng & Yun, 89]*
- (5) Minimum (false) promise *[Geelen'92]*

$\longrightarrow$ Dynamic variable/value orderings

Algorithms

Identifier	Choice of Var	Choice of val
LD+1	least domain	Minimum cost
LD+2	–	Min. cruciality
LD+3	–	Max promise
FE+2/4	Max. critical	Min. cruciality
FE+3/5	Min. promise	Max. promise

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

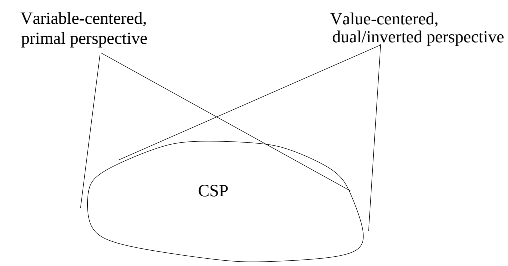
Implementation hack: domino effect, saves computations.

II- Permutation problems

$$\rightarrow \begin{cases} n = a \\ \text{constraint graph is complete} \\ \text{constraints are MUTEX, All-diffs} \end{cases}$$

→ matching, marriage problem
find one value for every variable and
exactly one variable for every value

Viewpoints: variable vs. values



→ Which viewpoint to take?
→ How to combine computations in viewpoints?

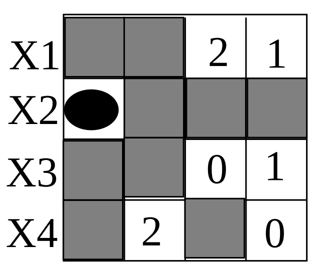
Permutation problems (cont'd)

At any point in BT,

future variables = # 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



PP: Full evaluation

Introduce: $\begin{cases} \text{LEFT}^{inv}(V_i | X_i = V_i) \\ \text{Promise}^{inv}(V_i = X_i) \end{cases}$

Combine viewpoints: CPromise$(X_i = V_i)$ min. $\begin{cases} \text{Promise}(X_i = V_i) \\ \text{Promise}^{inv}(V_i = X_i) \end{cases}$

Evaluate vars/vals using: $\begin{cases} \text{CPromise}(X_i) = \sum_k \text{CPromise}(X_i = V_k) \\ \text{CPromise}^{inv}(V_i) = \sum_k \text{CPromise}(X_k = V_i) \end{cases}$

Partial permutation problems ($n \leq a$)

→ fake a real PP with bogus variables

→ 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:
- 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

Algorithm	Average #backtrack	#backtrack-free solutions	Max. #bactrack
LD+formula 1	>45000	20	>2500000
LD+formula 2	>33000	15	>2500000
LD+formula 3	>1205	3	92379
LD+formula 1, dual	21.6	26	548
FE+formulae 2&4	9.5	71	812
FE+formulae 3&5	5.1	68	266
FP+formulae 2&4	5.6	81	496
FP+formulae 3&5	4.2	68	224
FE+formulae 3&5, dual	0.38	90	12

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.