BE-BB\((k)\): A Hybrid Method for Solving CSPs and COPs

Matti Järvisalo
matti.jarvisalo@hut.fi
April 15, 2004


Motivation (1/2)

- Most CSP solvers apply
  - search or
  - dynamic programming.

- Search:
  - Branch-and-bound (BB) in constraint optimization
  - Relation propagation in constraint satisfaction
  - Worst-case: explore whole search tree; exponential in \( n \)

- Dynamic programming:
  - Sequence of transformations reduce problem size
  - Bucket elimination (BE): basic step variable elimination
  - Worst-case exponential in arity of induced constraints

Outline

- Motivation
- CSPs and COPs
- Branch-and-Bound (BB)
- Bucket elimination (BE)
- Combining BE and BB: BE-BB\((k)\)

Motivation (2/2)

- Idea: Combine BB and BE → get best out of both worlds?
  - Apply variable elimination if induces constraints are of low arity
  - Controlled by parameter \( k \)
  - Else switch to search

- Solution BE-BB\((k)\)
  - worst-case time/space exponential in \( k \)

- Properties:
  - May boost search in constraint satisfaction, no worsening effect
  - Overwhelming advantage on some optimization tasks
CSPs Revisited

Constraint satisfaction problem (CSP):

- $X = \{x_1, \ldots, x_n\}$: set of variables
- $D = \{D_1, \ldots, D_n\}$: set of domains, where $x_i \in D_i$
- $C = \{R_1, \ldots, R_m\}$: set of constraints, where $R \in C$ is a relation over the scope $\text{var}(R) \subseteq X$
- Solution: assignment of values for each $x_i \in X$ from $D_i$ s.t. constraints in $C$ are satisfied
- Arity of a constraint $R$ is $|\text{var}(R)|$
- Arity of a CSP: $\max_{R \in C} \{|\text{var}(R)|\}$

COPs Revisited

Constraint optimization problem (COP): a CSP with two types of constraints

- Hard constraints (as in a CSP)
- Soft constraints (denoting preferences among tuples)
- Constraints are seen as cost functions
  - Returns for each tuple a non-negative cost
  - Hard constraints assign cost $0/\infty$ to allowed/forbidden tuples
- Weighted CSP (WCSP):
  - Minimize the objective function: the sum of all constraints $C = \{f_1, \ldots, f_m\}$

  $$f^*(X) = \sum_{j=1}^{m} f_j$$

Branch–and–Bound Revisited

A search schema for COP solving:

- Traverses the search tree defined by the problem
- Internal nodes: incomplete assignments
- Leaf nodes: complete assignments (optimal or not)
- Upkeeps upper (UB) and lower bounds (LB) for the best possible solution
  - If $UB \leq LB(t)$ for a partial assignment $t$, backtrack
- Basic step: branching

Lower Bound Computation

$$\text{LB}(t) = \sum_{f \in C} \min_{q} \{f(t, q)\},$$

where

- $t$: the current partial assignment,
- $\min_{q} \{f(t, q)\}$: minimum cost extension of $t$ to variables in $\text{var}(f)$ not assigned in $t$
- Time complexity: $O(m \cdot d^{r-1})$
- Reduce to $O(m \cdot d^s)$ by considering only constraints $f$ having at most $s$ uninstantiated variables in $\text{var}(f)$
Bucket Elimination

A *dynamic programming schema* for solving COPs

- A variable ordering $o$ is assumed
- Partitions $C$ into *buckets* $B_i$
- $B_i$ contains such constraints $f$ in which $x_i$ is the highest one in $\text{var}(f)$ according to $o$
- *Eliminates variables* one-by-one in descending order according to $o$
- Summarizes the effect by *generating an additional constraint*

Bucket Elimination (2/2)

- The additional constraint:

$$\text{elim}_i\left(\sum_{f \in B_i} f\right),$$

where

- $(f + g)(X) = f(X) + g(X)$ with scope $\text{var}(f) \cup \text{var}(g)$, and
- $(\text{elim}(f)(X) = \min_{a \in D_i} \{f(X[x_i = a])\}$ with scope $\text{var}(f) - \{x_i\}$

- Last elimination produces a constant function having the value of the optimal cost
- The optimal assignment can then be generated *backtrack-free*:

$$\sum_{f \in B_i} f$$

BE-BB($k$)

Worst-case comparison:

<table>
<thead>
<tr>
<th></th>
<th>time</th>
<th>space</th>
</tr>
</thead>
<tbody>
<tr>
<td>BB</td>
<td>exp in $n$</td>
<td>linear (in $n$)</td>
</tr>
<tr>
<td>BE</td>
<td>exp in arity of $f_i$, linear in $n$</td>
<td>exp in arity of $f_i$, linear in $n$</td>
</tr>
</tbody>
</table>

- Note: Determining the best ordering (w.r.t. $f_i$s) is *NP-complete*

BE-BB($k$): The following recursive idea

- Eliminate $x_i$s s.t. the arity of $f_i$ is $\leq k$ with BE
- Then apply BB to the reduced problem:
  - branch on a variable, then apply BE again if possible

Ending Remarks

BE-BB($k$)

- A generalization for CSP/COP solving of an idea of combining search and directed resolution in SAT solving
- Boosts branch-and-bound with bucket elimination
- A structural parameter defines when to BB/BE
- As usual, variable ordering is somewhat crucial
- “Overwhelming advantage” on some COPs reported