DM826 – Spring 2012

Modeling and Solving Constrained Optimization Problems

Lecture 9
Global Constraints

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

Outline

Declarative and Operational Semantic

- Declarative Semantic: specify **what** the constraint means. Evaluation criteria is expressivity.
- Operational Semantic: specify **how** the constraint is computed, i.e., is kept *consistent* with its declarative semantic. Evaluation criteria are efficiency and effectiveness.

Example

So far, we have defined only the Declarative Semantic of the `alldifferent` constraint, not its Operational Semantic.

Domain Consistency

Definition

A constraint C on the variables $x_1, \ldots, x_m$ with respective domains $D_1, \ldots, D_m$ is called domain consistent (or hyper-arc consistent) if for each variable x_i and each value $d_i \in D_i$ there exists compatible values in the domains of all the other variables of C, that is, there exists a tuple $(d_1, \ldots, d_i, \ldots, d_k) \in C$.

Consistency and Filtering Algorithms

- Different level of consistency (arc, bound, range, domain) are maintained by different filtering algorithms, which must be able to:
  1. Check consistency of C w.r.t. the current variable domains
  2. Remove inconsistent values from the variable domains

- The stronger is the level of consistency, the higher is the complexity of the filtering algorithm.

... again the `alldifferent` case

There exists in literature several filtering algorithms for the `alldifferent` constraints.

Domain consistency for alldifferent

1. build value graph $G = (X, D(X), E)$
2. compute maximum matching M in G
3. **if** $|M| < |X|$ **then return false**
4. mark all arcs in G_M that are not in M as unused
5. compute SCCs in G_M and mark all arcs in a SCC as used
6. perform breadth-first in G_M search starting from M-free vertices, and mark all traversed arcs as used if they belong to an even path
7. **for all** arcs (x_i, d) in G_M marked as unused **do**
   $D(x_i) := D(x_i) \setminus d$
   if$D(x_i) = \emptyset$ **then return false**
8. **return true**

Overall complexity: $O(n\sqrt{m} + (n + m) + m)$
It can be updated incrementally if other constraints remove some values.

Relaxed Consistency

Definition

A constraint C on the variables $x_1, \ldots, x_m$ with respective domains $D_1, \ldots, D_m$ is called bound(Z) consistent if for each variable x_i and each value $d_i \in \{\min(D_i), \max(D_i)\}$ there exists compatible values between the min and max domain of all the other variables of C, that is, there exists a value $d_j \in [\min(D_i), \max(D_i)]$ for all $j \neq i$ such that $(d_1, \ldots, d_i, \ldots, d_k) \in C$.

Definition

A constraint C on the variables $x_1, \ldots, x_m$ with respective domains $D_1, \ldots, D_m$ is called range consistent if for each variable x_i and each value $d_i \in D_i$ there exists compatible values between the min and max domain of all the other variables of C, that is, there exists a value $d_j \in [\min(D_i), \max(D_i)]$ for all $j \neq i$ such that $(d_1, \ldots, d_i, \ldots, d_k) \in C$.

Bound Consistency [Mehlorn&Thiel2000]

Definition (Convex Graph)

A bipartite graph $G = (X, Y, E)$ is convex if the vertices of Y can be assigned distinct integers from $[1, |Y|]$ such that for every vertex $x \in X$, the numbers assigned to its neighbors form a subinterval of $[1, |Y|]$.

In convex graph we can find a matching in linear time.

Survey of complexity: effectiveness and efficiency

Consistency	Idea	Complexity	Amort.	Reference(s)
arc		$O(n^2)$		[VanHentenryck1989]
bound	Hall	$O(n \log n)$		[Puget1998]
	Flows			[Mehlhorn&Thiel2000]
	Hall			[Lopez&All2003]
		$O(n)$		[Mehlhorn&Thiel2000]
				[Lopez&All2003]
range	Hall	$O(n^2)$		[Leconte1996]
domain	Flows	$O(n\sqrt{m})$	$O(n\sqrt{k})$	[Régin1994],[Costa1994]

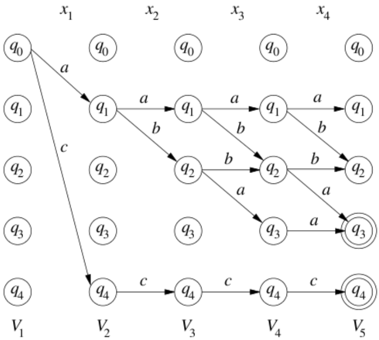
Where $n =$ number of variables, $m = \sum_{i \in 1...n} |D_i|$, and $k =$ number of values removed.

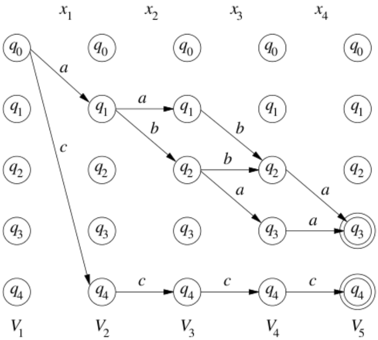
Filtering

gcc

Filtering

`regular`

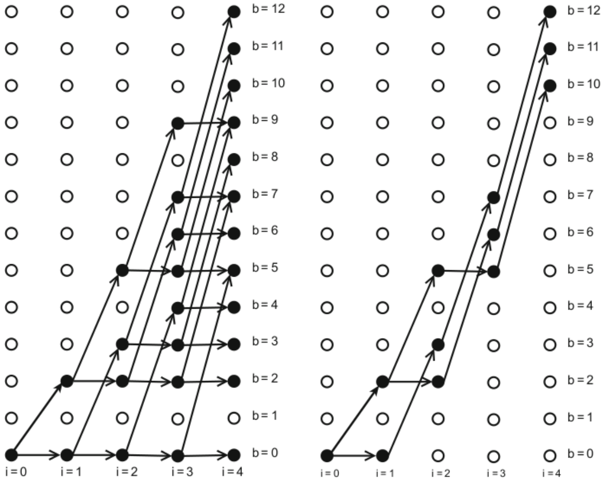




Filtering

subsetsum

$10 \leq 2x_1 + 3x_2 + 4x_3 + 5x_4 \leq 12$



Filtering

`disjunctive`
`cumulative`

Filtering

Exercise:
`linear`
`element`

Filtering Algorithm Design

1. Filtering algorithms based on a generic algorithm
   Simple "square" constraint using `element`:

$$\texttt{element}(y, [2, 4, 8, 16, 32], x), x \in \{1, 2, 3, 4, 5\}$$

2. Filtering algorithms based on existing algorithms
   Reuse existing algorithms for filtering (e.g., flows algorithms, dynamic programming).

3. Filtering algorithms based on ad-hoc algorithms
   Pay particular attention to incrementality and amortized complexity
4. Filtering algorithms based on model reformulation
   See the Constraint Decomposition approach

Outline

1. Global Constraints

2. Soft Constraints

3. Optimization Constraints

Soft Constraints

Soft constraint

A *soft constraint* is a constraint that may be violated. We measure the violation of each constraint, and the goal is to minimize the total amount of violation of all soft-constraints.

Definition

A *violation measure* for a soft-constraint $C(x_1, \ldots, x_n)$ is a function

$$\mu : D(x_1) \times \cdots \times D(x_n) \rightarrow \mathbb{Q}.$$

This measure is represented by a *cost* variable z.

Violation measures

- The variable-based violation measure μ_{var} counts the minimum number of variables that need to change their value in order to satisfy the constraint.
- The decomposition-based violation measure μ_{dec} counts the number of constraints in the binary decomposition that are violated.

The `soft-alldifferent`

Definition

Let $x_1, x_2, ..., x_n, z$ be variables with respective finite domains $D(x_1), D(x_2), ..., D(x_n), D(z)$. Let μ be a violation measure for the `alldifferent` constraint. Then

$$\texttt{soft-alldifferent}(x_1, ..., x_n, z, \mu) = \{(d_1, ..., d_n, d) \mid \forall i.d_i \in D(x_i), d \in D(z), \mu(d_1, ..., d_n) \leq d\}$$

is the soft alldifferent constraint with respect to μ.

The soft-alldifferent: an example

Example

Consider the following CSP

$$x_1 \in \{a, b\}, x_2 \in \{a, b\}, x_3 \in \{a, b\}, x_4 \in \{a, b, c\}, z \in \mathbb{Z}^+$$
$$\texttt{soft-alldifferent}(x_1, x_2, x_3, x_4, \mu, z)$$
$$\min z$$

We have for instance $\mu_{var}(b, b, b, b) = 3$ and $\mu_{dec}(b, b, b, b) = 6$.

`balancing`

Outline

1. Global Constraints

2. Soft Constraints

3. Optimization Constraints

Optimization Constraints

Optimization Constraint bring the costs of variable-value pair into the declarative semantic of the constraints.

The filtering does take into account the cost, and a tuple may be inconsistent because it does not lead to a solution of "at least" a given cost.

gcc with costs

cardinality or cost_gcc (global cardinality constraint with costs)

Let $x_1, \ldots, x_n$ be assignment variables whose domains are contained in $\{v_1, \ldots, v_{n'}\}$ and let $\{c_{v_1}, \ldots, c_{v_{n'}}\}$ be count variables whose domains are sets of integers and $w(x, d) \in Q$ are costs. Then

$$
\begin{aligned}
\texttt{cost\_gcc}([x_1, ..., x_n], [c_{v_1}, ..., c_{v_{n'}}], z, w) = \\
& \{(d_1, ..., d_n, o_1, ..., o_{n'}) \mid \\
& \{(d_1, ..., d_n, o_1, ..., o_{n'}) \in \text{gcc}(([x_1, ..., x_n], [c_{v_1}, ..., c_{v_{n'}}]), \\
& \forall d_j \in D(x_j)\, d \in D(z) \sum_i w(x_i, d_i) \le d\}.
\end{aligned}
$$

Reduced-Cost Based Filtering [Focacci&al1999]

Definition

Let $X = \{x_1, ..., x_n\}$ be a set of variables with corresponding finite domains $D(x_1), ..., D(x_n)$. We assume that each pair (x_i, j) with $j \in D(x_i)$ induces a cost c_{ij}.
We extend any global constraint C on X to an optimization constraint opt_C by introducing
a cost variable z (that we wish to minimize) and defining

$$\texttt{opt\_C}(x_1, ..., x_n, z, c) = \{(d_1, ..., d_n, d) | (d_1, ..., d_n) \in C(x_1, ..., x_n),$$

$$\forall i.d_i \in D(x_i), d \in D(z), \sum_{i=1,...,n} c_{id_i} \leq d\}.$$

Linear Relaxation

We introduce binary variables y_{ij} for all $i \in \{1, ..., n\}$ and $j \in D(x_i)$, such that

$$x_i = j \Leftrightarrow y_{ij} = 1, \qquad \forall i = 1, \ldots, n, \forall j \in D(x_i),$$

$$x_i \neq j \Leftrightarrow y_{ij} = 0, \qquad \forall i = 1, \ldots, n, \forall j \in D(x_i) \sum_{j \in D(x_i)} y_{ij}$$

+ constraint dependent linear equation

The reduced-cost are given w.r.t. the objective:

$$\sum_{i=1,\ldots,n} \sum_{j \in D(x_i)} c_{ij} y_{ij}$$

Example

`alldiff`

$$
\begin{array}{ll}
\min & \sum_{i,j} c_{i,j} y_{i,j} \\
& \sum_{j \in D(x_i)} y_{ij} = 1, \quad \forall i = 1, \ldots, n \\
& \sum_{i=1,\ldots,n} y_{ij} \leq 1, \quad \forall j \in D(x_i) \\
& y_{ij} \leq 0
\end{array}
$$

Filtering by Reduced-Cost (aka "variable fixing")

Recall that reduced-costs estimate the increase of the objective function when we force a variable into the solution.

Let $\bar{c}_{ij}$ be the reduced cost for the variable-value pair $x_i = j$, and let z^* be the optimal value of the current linear relaxation.

We apply the following filtering rule:

$$\textbf{if } z^* + \bar{c}_{ij} > \max D(z) \textbf{ then } D(x_i) \leftarrow D(x_i) \setminus \{j\}.$$

References

Régin J.C. (2011). **Global constraints: A survey**. In *Hybrid Optimization*, edited by P.M. Pardalos, P. van Hentenryck, and M. Milano, vol. 45 of **Optimization and Its Applications**, pp. 63–134. Springer New York.

van Hoeve W. and Katriel I. (2006). **Global constraints**. In *Handbook of Constraint Programming*, chap. 6. Elsevier.