Combinatorial Decision Making and Optimization

Last update: 29 February 2024

Academic Year 2023 – 2024
Alma Mater Studiorum · University of Bologna

Contents

1 Constraint programming

1.1 Definitions

Constraint satisfaction problem (CSP) Triple $\langle X, D, C \rangle$ where:

Constraint satisfaction problem (CSP)

- X is the set of decision variables $\{x_1, \dots, x_n\}$.
- D is the set of domains $\{D(X_1), \dots, D(X_n)\}$ or $\{D_1, \dots, D_n\}$ for X.
- C is the set of constraints $\{C_1, \dots, C_m\}$. Each C_i is a relation over the domain of X (i.e. $C_i \subseteq D_j \times \dots \times D_k$).

Constraint optimization problem (COP) Tuple $\langle X, D, C, f \rangle$ where $\langle X, D, C \rangle$ is a CSP and f is the objective variable to minimize or maximize.

Constraint optimization problem (COP)

Constraint

Constraint

Extensional representation List all allowed combinations.

Intensional representation Declarative relations between variables.

Symmetry Search states that lead to the same result.

Variable symmetry A permutation of the assignment order of the variables results in the same feasible or unfeasible solution.

Variable symmetry

Value symmetry A permutation of the values in the domain results in the same feasible or unfeasible solution.

Value symmetry

Remark. Variable and value symmetries can be combined resulting in a total of $2n!$ possible symmetries.

1.2 Modeling techniques

Auxiliary variables Add new variables to capture constraints difficult to model or to reduce the search space by collapsing multiple variables into one.

Auxiliary variables

Global constraints Relation between an arbitrary number of variables. It is usually computationally faster than listing multiple constraints.

Global constraints

Implied constraints Semantically redundant constraints with the advantage of pruning the search space earlier.

Implied constraints

Remark. A purely redundant constraint is also an implied constraint but it does not give any computational improvement.

Symmetry breaking constraints Constraints to avoid considering symmetric states. Usually, it is sufficient to fix an ordering of the variables.

Symmetry breaking constraints

Remark. When introducing symmetry breaking constraints, it might be possible to add new simplifications and implied constraints.

Dual viewpoint Modeling a problem from a different perspective might result in a more efficient search. Dual viewpoint

Example. Exploiting geometric symmetries.

Combined model Merging two or more models of the same problem by adding channeling constraints to guarantee consistency. Combined model

Combining two models can be useful for obtaining the advantages of both (e.g. one model uses global constraints, while the other handles symmetries).

Remark. When combining multiple models, some constraints might be simplified as one of the models already captures it natively.

1.3 Constraints

1.3.1 Local consistency

Examine individual constraints and detect inconsistent partial assignments.

Generalized arc consistency (GAC) Generalized arc consistency (GAC)

Support Given a constraint defined on k variables $C \subseteq D(X_1) \times \cdots \times D(X_k)$, each tuple $(d_1, \dots, d_k) \in C$ (i.e. allowed variables assignment) is a support for C.

A constraint $C(X_1, \dots, X_k)$ is GAC (GAC(C)) iff:

$$\forall X_i \in \{X_1, \dots, X_k\}, \forall v \in D(X_i) : v \text{ belongs to a support for } C$$

Remark. A CSP is GAC when all its constraints are GAC.

Example (Generalized arc consistency)**.** Given the variables $D(X_1) = \{1, 2, 3\}$, $D(X_2) = \{1, 2\}$, $D(X_3) = \{1, 2\}$ and the constraint $C : \texttt{alldifferent}([X_1, X_2, X_3])$, C is not GAC as $1 \in D(X_1)$ and $2 \in D(X_2)$ do not have a support.

By applying a constraint propagation algorithm, we can reduce the domain to: $D(X_1) = \{\cancel{1}, \cancel{2}, 3\}$ and $D(X_2) = D(X_3) = \{1, 2\}$. Now C is GAC.

Arc consistency (AC) A constraint is arc consistent when its binary constrains are GAC. Arc consistency (AC)

Example (Arc consistency)**.** Given the variables $D(X_1) = \{1, 2, 3\}$, $D(X_2) = \{2, 3, 4\}$ and the constraint $C : X_1 = X_2$, C is not arc consistent as $1 \in D(X_1)$ and $4 \in D(X_2)$ do not have a support.

By applying a constraint propagation algorithm, we can reduce the domain to: $D(X_1) = \{\cancel{1}, 2, 3\}$ and $D(X_2) = \{2, 3, \cancel{4}\}$. Now C is arc consistent.

Bounds consistency (BC) Can be applied on totally ordered domains. The domain of a variable X_i is relaxed from $D(X_i)$ to the interval $[\min\{D(X_i)\}, \max\{D(X_i)\}]$. Bounds consistency (BC)

Bound support Given a constraint defined on k variables $C(X_1, \dots, X_k)$, each tuple $(d_1, \dots, d_k) \in C$, where $d_i \in [\min\{D(X_i)\}, \max\{D(X_i)\}]$, is a bound support for C.

A constraint $C(X_1, \dots, X_k)$ is BC (BC(C)) iff:

$$\begin{aligned} &\forall X_i \in \{X_1, \dots, X_k\} : \\ &\qquad \min\{D(X_i)\} \text{ and } \max\{D(X_i)\} \text{ belong to a bound support for } C \end{aligned}$$

Remark. BC might not detect all GAC inconsistencies, but it is computationally cheaper.

Remark. On monotonic constraints, BC and GAC are equivalent.

Example. Given the variables $D(X_1) = D(X_2) = D(X_3) = \{1, 3\}$ and the constraint C : $\texttt{alldifferent}([X_1, X_2, X_3])$, C is BC as all $\min\{D(X_i)\}$ and $\max\{D(X_i)\}$ belong to the bound support $\{(d_1, d_2, d_3) \mid d_i \in [1, 3] \land d_1 \neq d_2 \neq d_3\}$

On the other hand, C fails with GAC.

1.3.2 Constraint propagation

Constraint propagation Algorithm that removes values from the domains of the variables to achieve a given level of consistency. Constraint propagation

Constraint propagation algorithms interact with each other and already propagated constraints might be woke up again by another constraint. Propagation will eventually reach a fixed point.

Specialized propagation Propagation algorithm specific to a given constraint. Allows to exploit the semantics of the constraint for a generally more efficient approach. Specialized propagation

1.3.3 Global constraints

Constraints to capture complex, non-binary, and recurring features of the variables. Usually, global constraints are enforced using specialized propagation algorithms.

Counting constraints

Constrains the number of variables satisfying a condition or the occurrences of certain values. Counting constraints

All-different Enforces that all variables assume a different value.

$$\texttt{alldifferent}([X_1, \dots, X_k]) \iff \forall i, j, \in \{1, \dots, k\}, i \neq j : X_i \neq X_j$$

Global cardinality Enforces the number of times some values should appear among the variables.

$$\texttt{gcc}([X_1, \dots, X_k], [v_1, \dots, v_m], [O_1, \dots, O_m]) \iff \\ \forall j \in \{1, \dots, m\} : |\{X_i \mid X_i = v_j\}| = O_j$$

Among Constrains the number of occurrences of certain values among the variables.

$$\texttt{among}([X_1, \dots, X_k], \{v_1, \dots, v_n\}, l, u) \iff l \leq |\{X_i \mid X_i \in \{v_1, \dots, v_n\}\}| \leq u$$

Sequencing constraints

Enforces a pattern on a sequence of variables. Sequencing constraints

Sequence Enforces the number of times certain values can appear in a subsequence of a given length q.

$$\texttt{sequence}(l, u, q, [X_1, \dots, X_k], \{v_1, \dots, v_n\}) \iff \\ \forall i \in [1, \dots, k - q + 1] : \texttt{among}([X_1, \dots, X_{i+q-1}], \{v_1, \dots, v_n\}, l, u)$$

Scheduling constraints

Useful to schedule tasks with release times, duration, deadlines, and resource limitations. Scheduling constraints

Disjunctive resource Enforces that the tasks do not overlap over time. Given the start time S_i and the duration D_i of k tasks:

$$\texttt{disjunctive}([S_1, \dots, S_k],[D_1, \dots, D_k]) \iff \forall i < j : (S_i + D_i \leq S_j) \vee (S_j + D_j \leq S_i)$$

Cumulative resource Constrains the usage of a shared resource. Given a resource with capacity C and the start time S_i, the duration D_i, and the resource requirement R_i of k tasks:

$$\texttt{cumulative}([S_1, \dots, S_k],[D_1, \dots, D_k],[R_1, \dots, R_k], C) \iff \forall u \in \{D_1, \dots, D_k\} : \sum_{i \text{ s.t. } S_i \leq u < S_i + D_i} R_i \leq C$$

Ordering constraints

Enforce an ordering between variables or values. Ordering constraints

Lexicographic ordering Enforces that a sequence of variables is lexicographically less than or equal to another sequence.

$$\begin{aligned} \texttt{lex}\leq([X_1, \dots, X_k],[Y_1, \dots, Y_k]) \iff & X_1 \leq Y_1 \wedge \\ & (X_1 = Y_1 \Rightarrow X_2 \leq Y_2) \wedge \\ & \cdots \wedge \\ & ((X_1 = Y_1 \wedge \dots \wedge X_{k-1} = Y_{k-1}) \Rightarrow X_k \leq Y_k) \end{aligned}$$

Generic purpose constraints

Define constraints in an extensive way. Generic purpose constraints

Table Associate to the variables their allowed assignments.

Specialized propagation

Constraint decomposition A global constraint is decomposed into smaller and simpler constraints with known propagation algorithms. Constraint decomposition

Remark. As the problem is decomposed, some inconsistencies may not be detected.

Example. (Decomposition of `among`) The `among` constraint can be decomposed as follows:

Variables B_i with $D(B_i) = \{0, 1\}$ for $1 \leq i \leq k$.

Constraints

- $C_i : B_i = 1 \iff X_i \in v$ for $1 \leq i \leq k$.
- $C_{k+1} : \sum_i B_i = N$.

AC(C_i) and BC(C_{k+1}) ensures GAC on `among`.

Dedicated propagation algorithm Ad-hoc algorithm that implements an efficient propagation.

Dedicated propagation algorithm

Example. (`alldifferent` through maximal matching)

Bipartite graph Graph where the edges are partitioned in two groups U and V. Nodes in U can only connect to nodes in V.

Maximal matching Largest subsets of edges such that there are no edges with nodes in common.

Define a bipartite graph $G = (U \cup V, E)$ where:

- $U = \{X_1, \dots, X_k\}$ are the variables.
- $V = D(X_1) \cup \dots \cup D(X_k)$ are the possible values of the variables.
- $E = \{(X_i, v) \mid X_i \in U, v \in V : v \in D(X_i)\}$ contains the edges that connect every variable in U to its possible values in V.

All the possible variable assignments of $X_1, \dots, X_k$ are the maximal matchings in G.