

Combinatorial Problems

- n variables, k values $\leadsto k^n$ alternatives
- Constraint satisfaction and constrained optimization
- Many industrial applications in classical **operations research** areas, e.g.
 - Scheduling
 - Resource allocation
 - Personnel planning
 - Transportation
- Many other application areas. e.g.
 - Computational physics
 - Computational biology

Branch-and-Infer: A Unifying Framework for Integer and Finite Domain Constraint Programming

Alexander Bockmayr
Université Henri Poincaré & LORIA
Nancy, France

Joint work with Thomas Kasper

Two General Approaches

- **Integer linear programming (ILP)**
 - Classical approach in operations research
 - Based on linear programming \leadsto Simplex method
 - Studied for more than 40 years
 - Many impressive results in various application areas
- **Finite domain constraint programming (CP(\mathcal{FD}))**
 - Promising new approach
 - Started about 10 years ago as a combination of logic programming and constraint satisfaction \leadsto CHIP system
 - Strong results in areas where traditional operations research is weak, e.g. scheduling.

Questions

1. What is the relationship between the two approaches?
 - Model building – Language
 - Model solving – Algorithms \leadsto **Branch-and-infer**
2. How can we combine them ?
 - Symbolic constraints in ILP
 - Cooperation of ILP and CP(\mathcal{FD})

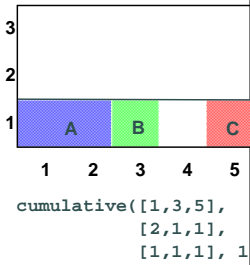
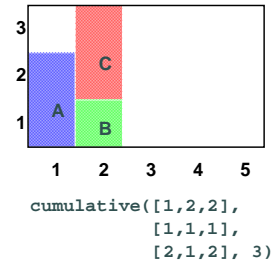
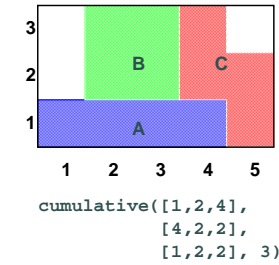
Languages and Algorithms

	ILP	CP(FD)
Language	Linear arithmetic —	Arithmetic constraints Symbolic constraints
Algorithms	Global consistency (LP)	Local consistency
	Cutting planes	Domain reduction
	Branch-and-bound Branch-and-cut	User-defined enumeration

Symbolic constraints \leadsto more expressivity + more efficiency

Examples of Symbolic Constraints

- `alldifferent`($[X_1, \dots, X_n]$)
- `cumulative`($[S_1, \dots, S_n], [D_1, \dots, D_n], [R_1, \dots, R_n], L$).
 - n tasks with starting time S_i , duration D_i , resource demand R_i
 - Global resource limit L , may never be exceeded.



Bin Packing

- Pack n items of size $g_i, i = 1, \dots, n$, into (at most) n bins of capacity c .
- Put first m items into different bins.
- Find minimal number of bins necessary.

Integer Linear Programming

- 0-1 variables x_{ij} – item i is packed in bin j
- 0-1 variables y_j – bin j is needed

$$\begin{aligned}
 \text{Minimize} \quad & y_1 + \dots + y_n \\
 \text{s.t.} \quad & x_{i1} + \dots + x_{in} = 1, \quad i = 1, \dots, n \\
 & g_1 x_{1j} + \dots + g_n x_{nj} \leq c y_j, \quad j = 1, \dots, n \\
 & x_{1j} + \dots + x_{mj} \leq 1, \quad j = 1, \dots, n \\
 & x_{ij}, y_j \in \{0, 1\}, \quad i, j = 1, \dots, n
 \end{aligned}$$

- $n^2 + n$ variables over domain $\{0, 1\}$, $3n$ constraints

Constraint Program

- Domain variables $z_i \in \{1, \dots, n\}$ – bin for item i
- Domain variables $z \in \{1, \dots, n\}$ – number of bins needed

$$\begin{aligned} & \text{Minimize } z \\ & \text{s.t. } \text{cumulative}([z_1, \dots, z_n], \\ & \quad [1, \dots, 1], \\ & \quad [g_1, \dots, g_n], c, z), \\ & \quad \text{alldifferent}([z_1, \dots, z_m]), \\ & \quad z_1, \dots, z_n, z \in \{1, \dots, n\} \end{aligned}$$

- $n + 1$ variables over domain $\{1, \dots, n\}$, 2 constraints

Constraint Classification

- Arithmetic constraints

$$\sum_{i=1}^n a_i x_i \diamond b, \quad a_i, b \in \mathbb{Q}, \quad \diamond \in \{\leq, \geq, =, \neq, >, <\}$$

- Integrality constraints

$$\text{integer}([x_1, \dots, x_n])$$

- Symbolic constraints

$$\text{alldifferent}([x_1, \dots, x_n]), \text{cumulative}(\dots), \dots$$

Primitive and Non-Primitive Constraints

Partition constraint language: $L = \text{Prim}(L) \cup \text{NPrim}(L)$

- $\text{Prim}(L)$, primitive constraints, easy to solve
- $\text{NPrim}(L)$, non-primitive constraints, difficult to solve

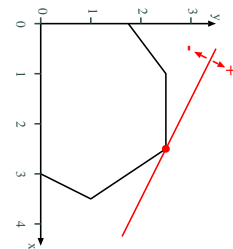
	Primitive	Non-Primitive
ILP	$\sum a_i x_i \diamond b, \diamond \in \{=, \leq, \geq\}$	integer
CP(FD)	$x \leq u, x \geq l, x \neq v, x = y$ integer	$\sum a_i x_i \diamond b, \diamond \in \{=, \leq, \geq, <, >, \neq\}$ alldifferent, cumulative, ...

Primitive Constraints in ILP

- Linear equations and inequalities

$$\begin{aligned} a_{11}x_1 + \dots + a_{1n}x_n &\leq b_1 \\ \vdots &\vdots \\ a_{m1}x_1 + \dots + a_{mn}x_n &\leq b_m \\ x_1, \dots, x_n &\in \mathbb{R} \end{aligned} \iff \begin{aligned} Ax &\geq b \\ A &\in \mathbb{R}^{m \times n} \\ b &\in \mathbb{R}^m \\ x &\in \mathbb{R}^n \end{aligned}$$

- $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ polyhedron

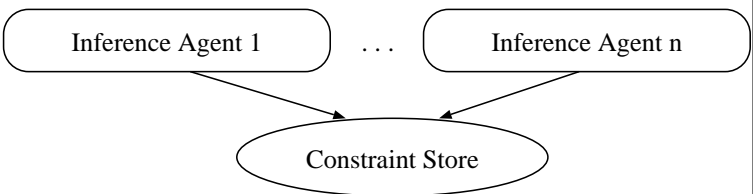


- Linear optimization

$$\max \{c^T x \mid Ax \leq b, x \in \mathbb{R}^n\}$$

Computational Architecture

- Primitive constraints \rightsquigarrow **constraint store**
- Non-primitive constraints communicate through the store, i.e. the primitive constraints.

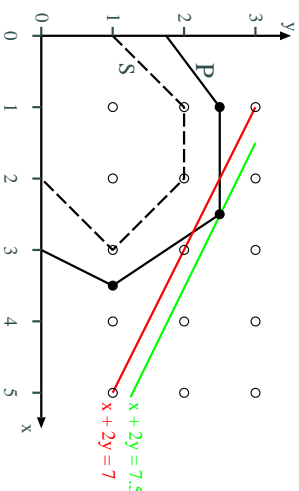


\rightsquigarrow more expressive primitive constraints \rightarrow more communication

Cutting Plane Principles

- **Chvátal-Gomory cutting plane**

$$C-G: \frac{Ax \geq b}{w^T A x \geq \lceil w^T b \rceil} \quad \text{if } \begin{matrix} w \geq 0, \\ w^T A \in \mathbb{Z}^n \end{matrix}$$



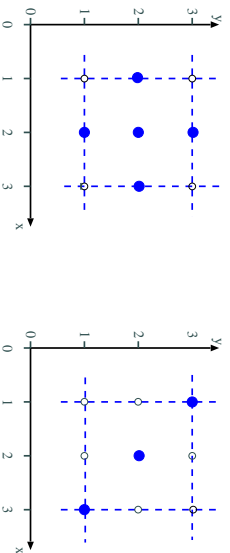
- **Disjunctive cutting plane**

$$DP: \frac{Ax \geq b \vee Cx \geq d}{\max(u^T A, v^T C) x \geq \min(u^T b, v^T d)} \quad \text{if } \begin{matrix} u \geq 0, \\ v \geq 0 \end{matrix}$$

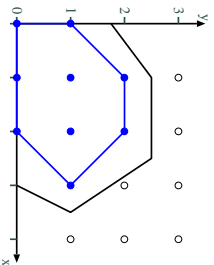
\rightsquigarrow lift-and-project method

Problem Solving

- Reduce non-primitive to primitive constraints \rightsquigarrow solve **relaxation**
- Cannot replace a non-primitive constraint NP by an equivalent set of primitive constraints P
- **CP(FD)**: P may not exist



- **ILP**: P exists, but in general cannot be computed



- **Infer some primitive constraints** \rightsquigarrow tighten the relaxation

Inference in ILP and CP(FD)

	CP(FD)	ILP
Inferences	$x \geq l, x \leq u$ Bound reasoning $x \neq v$ Domain reasoning	$\sum a_i x_i \leq b$ Cutting planes
Relaxation	Discrete - \mathbb{Z}	Continuous - \mathbb{Q}
Algorithms	Local consistency Propagation	Linear programming Separation

Branch-and-Infer

- Computation states $\langle P, S \rangle$, where P is a set of constraint problems, S a set of solutions.

- Infer

$$\text{bi_infer: } \frac{\langle (c \wedge C) . P, S \rangle}{\langle (p \wedge (c \wedge C)) . P, S \rangle} \quad \text{if } \begin{array}{l} c \text{ is non-primitive,} \\ p \text{ is primitive,} \\ \text{Prim}(C) \wedge c \rightarrow p, \\ \text{Prim}(C) \not\rightarrow p. \end{array}$$

- Branch

$$\text{bi_branch: } \frac{\langle C . P, S \rangle}{\langle (c_1 \wedge C) . \dots . (c_k \wedge C) . P, S \rangle} \quad \text{if } \begin{array}{l} C \equiv C \wedge (\bigvee_{i=1}^k c_i) \\ c_i \text{ primitive} \\ \text{Prim}(C) \not\rightarrow c_i \end{array}$$

- Unifying framework for ILP and CP(\mathcal{FD}) (Bockmayr/Kasper 98)

Pruning the Enumeration Tree

- Unsatisfiability of the relaxation

$$\text{bi_clash: } \frac{\langle C . P, S \rangle}{\langle P, S \rangle} \quad \text{if } \text{Prim}(C) \rightarrow \perp$$

- Lower and upper bounds

- $\max\{f(x) \mid x \in \text{Sol}(C)\}$
- Feasible solution $s^* \in \text{Sol}(C) \rightsquigarrow$ **lower bound** $f(s^*)$
- Relaxation $\max\{f(x) \mid x \in \text{Sol}(\text{Prim}(C))\} \rightsquigarrow$ **upper bound**

Branch-and-Bound, Branch-and-Relax, Branch-and-Cut

- CP(\mathcal{FD}): Only lower bounds \rightsquigarrow Branch-and-Bound

$$\text{bi_climb: } \frac{\langle C . P, \{s\} \rangle}{\langle (c \wedge C) . (c \wedge P), \{s^*\} \rangle} \quad \text{if } \begin{array}{l} s^* = \text{extract}(\text{Prim}(C)) \\ f(s^*) > f(s) \\ c \equiv [f(x) \geq f(s^*) + 1]. \end{array}$$

- ILP: Lower and upper bounds \rightsquigarrow Branch-and-Relax

$$\text{bi_bound: } \frac{\langle C . P, \{s\} \rangle}{\langle P, \{s\} \rangle} \quad \text{if } \max\{f(x) \mid x \in \text{Sol}(\text{Prim}(C))\} \leq f(s)$$

Tighten the relaxation using cutting planes \rightsquigarrow Branch-and-Cut

(= bi_bound + bi_infer)

Application I: Symbolic Constraints in ILP

- Transfer global constraint concept from CP(\mathcal{FD}) to ILP
 \rightsquigarrow new non-primitive constraints in ILP
- Expressiveness: **Abstract** definition of a set $S \subseteq \mathbb{Q}^n$, without giving explicit an equivalent set C of linear constraints
 - C does not exist, e.g. S is not convex.
 - C is not known, e.g. S is defined by a non-linear constraint.
 - C is known, but too large, e.g. traveling salesman problem.
- Efficiency: **Compact** representation of known classes of arithmetic constraints
 - Make explicit structural information in the model.
 - Integrate specialized cutting plane algorithms into a general solver.

Example: Warehouse Location

- Assign m customers to n warehouses
- f_j the fixed cost of warehouse j
- v_{ij} the variable cost for customer i and warehouse j

$$\begin{aligned} \min \quad & \sum_{i=1}^m \sum_{j=1}^n v_{ij} x_{ij} + \sum_{j=1}^n f_j y_j \\ \text{s.t.} \quad & \text{assign}([x_{11}, \dots, x_{1n}], \dots, [x_{m1}, \dots, x_{mn}]), \\ & [[1, \dots, 1], \dots, [1, \dots, 1]], \\ & [m, \dots, m], [y_1, \dots, y_n]), \\ & 0 \leq x_{ij} \leq 1, \quad 0 \leq y_j \leq 1, \\ & \text{integral}([x_{11}, \dots, x_{mn}, y_1, \dots, y_n]). \end{aligned}$$

assign Constraint

`assign(Assignments, Weights, Capacities, Indicators)`

- **Assignments:** A list of lists $[[x_{11}, \dots, x_{1n}], \dots, [x_{m1}, \dots, x_{mn}]]$, $m, n \geq 1$, of 0-1 variables x_{ij} or values.
- **Weights:** A list of lists $[[w_{11}, \dots, w_{1n}], \dots, [w_{m1}, \dots, w_{mn}]]$, $m, n \geq 1$, of non-negative rational numbers w_{ij} .
- **Capacities:** A list $[c_1, \dots, c_n]$ of non-negative rational numbers c_j .
- **Indicators:** A list $[y_1, \dots, y_n]$ of 0-1 variables or values.

Multiproduct Colocation Problem

- x_{ij}^p : client i receives product p from location j
- y_j : plant is opened at location j .
- z_j^p : machine for product p is installed at location j .

$$\begin{aligned} \min \quad & \sum_{p \in P} \sum_{i \in M} \sum_{j \in N} v_{ij}^p x_{ij}^p + \sum_{j \in N} f_j y_j + \sum_{p \in P} \sum_{j \in N} g_j^p z_j^p \\ \text{s.t.} \quad & \text{assign}([x_{11}^1, \dots, x_{1n}^1], \dots, [x_{m1}^1, \dots, x_{mn}^1]), \\ & [[1, \dots, 1], \dots, [1, \dots, 1]], [m, \dots, m], [z_1^1, \dots, z_n^1]), \\ & \text{assign}([x_{11}^2, \dots, x_{1n}^2], \dots, [x_{m1}^2, \dots, x_{mn}^2]), \\ & [[1, \dots, 1], \dots, [1, \dots, 1]], [m, \dots, m], [z_1^2, \dots, z_n^2]), \\ & \text{assign}([x_{11}^3, \dots, x_{1n}^3], \dots, [x_{m1}^3, \dots, x_{mn}^3]), \\ & [[1, \dots, 1], \dots, [1, \dots, 1]], [m, \dots, m], [z_1^3, \dots, z_n^3]), \\ & z_j^1 - y_j \leq 0, \quad z_j^2 - y_j \leq 0, \quad z_j^3 - y_j \leq 0, \quad j \in N \\ & \text{integral}([x_{11}^1, \dots, x_{mn}^3, y_1, \dots, y_n, z_1^1, \dots, z_n^3]) \end{aligned}$$

Weak and Strong Formulation

- Weak linear 0-1 model

$$\begin{aligned} \min \quad & \sum_{i \in M} \sum_{j \in N} v_{ij} x_{ij} + \sum_{j \in N} f_j y_j \\ \text{s.t.} \quad & \sum_{j \in N} x_{ij} = 1, \quad i \in M \\ & \sum_{i \in N} x_{ij} \leq m y_j, \quad j \in N \\ & 0 \leq x_{ij} \leq 1, \quad 0 \leq y_j \leq 1, \quad i \in M, j \in N \\ & \text{integral}([x_{11}, \dots, x_{mn}, y_1, \dots, y_n]). \end{aligned}$$

- Strong linear 0-1 model

$$\begin{aligned} \min \quad & \sum_{i \in M} \sum_{j \in N} v_{ij} x_{ij} + \sum_{j \in N} f_j y_j \\ \text{s.t.} \quad & \sum_{j \in N} x_{ij} = 1, \quad i \in M \\ & x_{ij} - y_j \leq 0, \quad i \in M, j \in N \\ & 0 \leq x_{ij} \leq 1, \quad 0 \leq y_j \leq 1, \quad i \in M, j \in N \\ & \text{integral}([x_{11}, \dots, x_{mn}, y_1, \dots, y_n]). \end{aligned}$$

Conclusion

- Branch-and-Infer
- Primitive vs. non-primitive constraints
- ILP: More powerful primitive constraints
- $CP(\mathcal{FD})$: More powerful non-primitive constraints
- Symbolic constraints in ILP
- Cooperating solvers

Application II: Cooperating Solvers

- **General idea**
 - Combine discrete and continuous constraint solving
 - Transfer information easily available in one approach to the other and vice versa
- **Integrate ILP in $CP(\mathcal{FD})$**
 - Primitive: $\text{Prim}(CP(\mathcal{FD}))$
 - Non-primitive: $\text{NPrim}(CP(\mathcal{FD})) \cup \{\text{linear}\}$
 - ↪ global treatment of inequalities
- **Integrate $CP(\mathcal{FD})$ in ILP**
 - Primitive: Prim(ILP)
 - Non-Primitive: $\text{NPrim(ILP)} \cup \text{NPrim}(CP(\mathcal{FD}))$
 - ↪ restrict inferences of $CP(\mathcal{FD})$ to $x \leq u, x \geq l, x = y$
- **Full merge**
 - Primitive: $\text{Prim(ILP)} \cup \{x \neq v\}$
 - Non-primitive: $\text{NPrim(ILP)} \cup \text{NPrim}(CP(\mathcal{FD}))$