Combining forces to solve Combinatorial Problems, a preliminary approach

Mohamed Siala, Emmanuel Hebrard, and Christian Artigues

LAAS-CNRS

Tarbes, France

Outline

Combinatorial Problems

Context

- Finite domain variables
- a fixed number of constraints over these variables

Combinatorial Problems

Context

- Finite domain variables
- a fixed number of constraints over these variables
- Is there a solution satisfying these constraints ?

Combinatorial Problems

Context

- Finite domain variables
- a fixed number of constraints over these variables
- Is there a solution satisfying these constraints ?

Combinatorial Problems

- The size of the search tree is exponential!
- There is no known algorithm for solving them in polynomial time
- NP-Complete/NP-Hard Problems

Constraint Satisfaction Problems

CSP

A constraint satisfaction problem (CSP) is a triplet $P = (\mathcal{X}, \mathcal{D}, \mathcal{C})$ where

- $\mathcal{X}$ is a set of variables.
- $\mathcal{D}$ is the related sets of values.
- $\mathcal{C}$ is a set of constraints.

A solution of a CSP is an assignment w satisfying all the constraints.

Constraint Satisfaction Problems

CSP

A constraint satisfaction problem (CSP) is a triplet $P = (\mathcal{X}, \mathcal{D}, \mathcal{C})$ where

- $\mathcal{X}$ is a set of variables.
- $\mathcal{D}$ is the related sets of values.
- $\mathcal{C}$ is a set of constraints.

A solution of a CSP is an assignment w satisfying all the constraints.

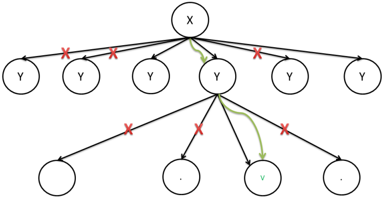
Example

- $X =< x, y >$
- $D =< \{1, 2, 3\}, \{4, 5\} >$
- $C_1 = \{x \text{ is even}\}$
- $C_2 = \{x + y = 6\}$

Propagation

- A propagator (or filtering algorithm) aims to remove some values that are inconsistent.
- Correctness & Checking

Figure: Propagation impact



Global constraints

- A global constraint is constraint over n variables.
- A global constraint captures a sub-problem.
- A global constraint can be used to solve different problems.
- A global constraint $\leftrightarrow$ specific propagator.

Propagation & Global Constraints ?

AllDifferent(X, Y, Z)

$X, Y, Z,\ D_X = D_Y = D_Z = \{1, 2\}$

Decomposition	Global Constraint
$C_1 : X \neq Y; C_2 : Y \neq Z; C_3 : Z \neq X;$	AllDifferent(X, Y, Z)
Propagate(C_1) : $C_1 : X \neq Y$ $D_X = D_Y\{1, 2\}$ $\rightarrow$ No propagation! Propagate(C_2),Propagate(C_3) : No propagation	$D_X = D_Y = D_Z = \{1, 2\}$ $\rightarrow$ Failure!

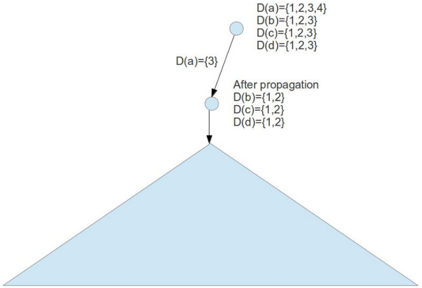
Learning in CP

- a,b,c,d integer variables pairwise different.
- $D(a) = \{1, 2, 3, 4\}$, $D(b) = \{1, 2, 3\}$, $D(c) = \{1, 2, 3\}$, $D(d) = \{1, 2, 3\}$
- $x_1, ..x_n$ n variables and $C_1, ..C_m$ m Constraints over these variables
- suppose that we branch on a, $x_1..x_n$, b, c, d

Learning in CP

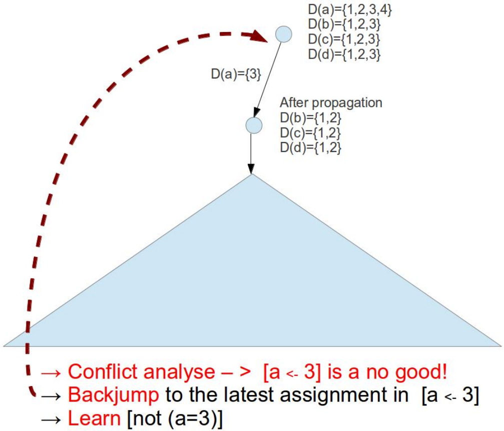
- a,b,c,d integer variables pairwise different.
- $D(a) = \{1,2,3,4\}$, $D(b) = \{1,2,3\}$, $D(c) = \{1,2,3\}$, $D(d) = \{1,2,3\}$
- $x_1, ..x_n$ n variables and $C_1, ..C_m$ m Constraints over these variables
- suppose that we branch on a, $x_1..x_n$, b, c, d

With a standard CP-Solver



Learning in CP

With learning :



Boolean Satisfiability (SAT)

A Sat-Problem

- Boolean variables
- CNF : a set of clauses (i.e. a set of disjunctions over these variables and their negations).
- For instance : $C \equiv (a \vee b) \wedge (\neg c \vee d \vee \neg e)$

Why SAT?

1. There is a community working on SAT-Problems!
2. Modern SAT-Solvers are able to deal with millions of variables and clauses

Satisfiability Modulo Theories

Suppose now that we want to solve :
$\phi \equiv ((x + y) = 32) \vee (a > 17)) \wedge ((w^3 + y = 0.53) \vee p_1 \vee \neg p_2)$

Satisfiability Modulo Theories

Suppose now that we want to solve :
$\phi \equiv ((x + y) = 32) \vee (a > 17)) \wedge ((w^3 + y = 0.53) \vee p_1 \vee \neg p_2)$
$\Rightarrow$ It looks like a CNF but ...

Satisfiability Modulo Theories

Suppose now that we want to solve :
$\phi \equiv ((x + y) = 32) \vee (a > 17)) \wedge ((w^3 + y = 0.53) \vee p_1 \vee \neg p_2)$

$\Rightarrow$ It looks like a CNF but ...

$\Rightarrow$ Satisfiability

Satisfiability Modulo Theories

Suppose now that we want to solve :
$\phi \equiv ((x + y) = 32) \vee (a > 17)) \wedge ((w^3 + y = 0.53) \vee p_1 \vee \neg p_2)$

$\Rightarrow$ It looks like a CNF but ...

$\Rightarrow$ Satisfiability Modulo Theories

Satisfiability Modulo Theories

Suppose now that we want to solve :
$\phi \equiv ((x + y) = 32) \vee (a > 17)) \wedge ((w^3 + y = 0.53) \vee p_1 \vee \neg p_2)$

$\Rightarrow$ It looks like a CNF but ...

$\Rightarrow$ Satisfiability Modulo Theories

$\Rightarrow$ First order formulas w.r.t some theories

Satisfiability Modulo Theories

Suppose now that we want to solve :
$\phi \equiv ((x + y) = 32) \vee (a > 17)) \wedge ((w^3 + y = 0.53) \vee p_1 \vee \neg p_2)$

$\Rightarrow$ It looks like a CNF but ...

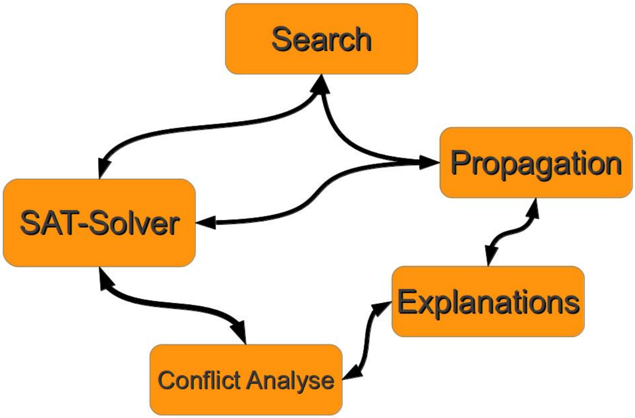
$\Rightarrow$ Satisfiability Modulo Theories

$\Rightarrow$ First order formulas w.r.t some theories

Lazy SMT

1. Exploiting SAT by abstracting the formula
2. Theory Propagation
3. Theory explanations for conflicts and propagation

Towards a hybrid solver



Definition

$\textsc{AtMostSeqCard}(u, q, d, [x_1, \ldots, x_n]) \Leftrightarrow$

$$\bigwedge_{i=0}^{n-q} \left(\sum_{l=1}^{q} x_{i+l} \leq u \right) \wedge \left(\sum_{i=1}^{n} x_i = d \right)$$

Definition

ATMOSTSEQCARD$(u, q, d, [x_1, \dots, x_n]) \Leftrightarrow$

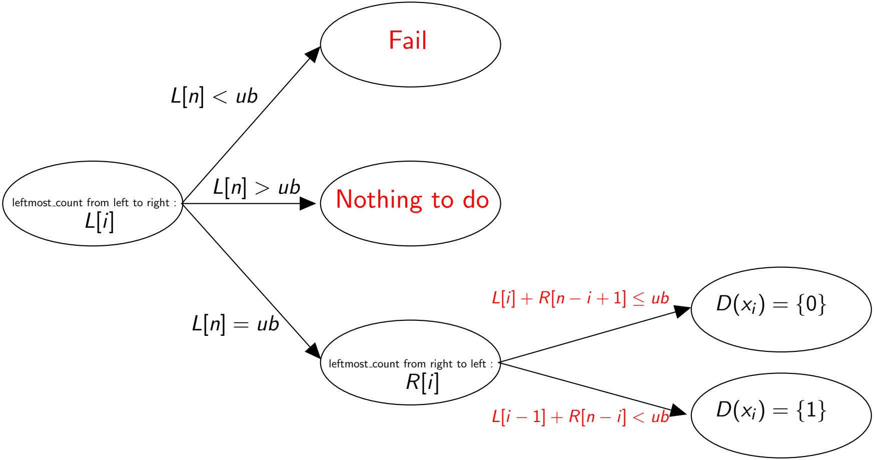
$$\bigwedge_{i=0}^{n-q} \left(\sum_{l=1}^{q} x_{i+l} \leq u\right) \wedge \left(\sum_{i=1}^{n} x_i = d\right)$$

Example ATMOSTSEQCARD$(2, 4, 4, [x_1, \dots, x_7])$

0 1 1 0 1 1 0

1 1 0 0 1 0 1

Filtering the Domains



Explaining the AtMostSeqCard constraint

Key idea

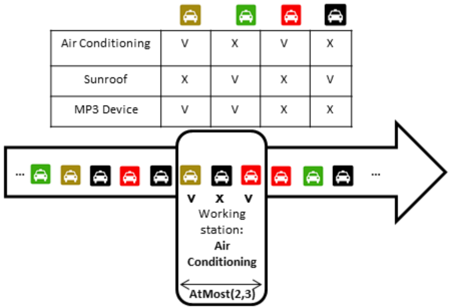
Let $S^{\cdot *}$ be a sequence defined as $\forall i \in [1, n]$, the domain of x_i in $S^{\cdot *}$ (denoted by $D^{\cdot *}(x_i)$) is defined as follows :

$$D^{\cdot *}(x_i) = \begin{cases} \{0,1\}, \text{ if } (D(x_i) = \{0\} \text{ and } max_i = u) \\ \{0,1\}, \text{ if } (D(x_i) = \{1\} \text{ and } max_i \neq u) \\ D(x_i) \text{ otherwise} \end{cases}$$

Theorem

Let $L^{\cdot *}$ the result of `leftmost_max` *on $S^{\cdot *}$.*
$\forall i \in [1, n]$, $L^{\cdot *}[i] = L[i]$.

Car-sequencing



Constraints

- Each class c is associated with a demand D_c.
- For each option j, each sub-sequence of size q_j must contain at most u_j cars requiring the option j.

Some results . . .

Easy Sat

	# solved	# TIME
mcp	368 / 368 100 %	0.17
hybrid	368 / 368 100 %	0.14
hybridSwitch	368 / 368 100 %	0.21
DefaultHybrid	368 / 368 100 %	0.33
sate2	368 / 368 100 %	3.15
sate3	368 / 368 100 %	3.01

Hard Sat

	# solved	# TIME
mcp	35 / 35 100%	16.72
hybrid	34 / 35 97%	3.05
hybridSwitch	34 / 35 97%	2.66
DefaultHybrid	16 / 35 45%	287.84
sate2	28 / 35 80%	289.32
sate3	31 / 35 88%	60.99

Some results . . .

Unsat instances

	# solved	# TIME
mcp	23 / 136 16%	300.55
hybrid	23 / 136 16%	300.55
hybridSwitch	36 / 136 26%	351.86
DefaultHybrid	35 / 136 25%	225.95
sate2	85 / 136 62%	92.45
sate3	66 / 136 48%	186.79

Current Contributions

- A linear time propagator for the AtMostSeqCard constraint
- Explaining the AtMostSeqCard constraint
- Getting started with the Hybrid solver

Future research

- Hybridisation & Hybridisation again ...
- Treating other problems (scheduling) in a SAT-CP context
- MiniZinc Challenge with a hybrid Solver
- Incremental SAT-Encoding for Finite Domain variables
- ...

Thank you!

Questions?