Solving hard sequencing problems via the AtMostSeqCard constraint

Mohamed Siala, Christian Artigues, Emmanuel Hebrard, Marie-José Huguet

ROC group : Recherche Opérationnelle/Optimisation Combinatoire/Contraintes

LAAS-CNRS

Bordeaux, France

Outline

Context & Background

The AtMostSeqCard constraint

AtMostSeqCard in a Hybrid CP/SAT context

Conclusion

Context

- (Discrete) Combinatorial Problems
- NP-Complete/NP-Hard Problems
- Constraint Satisfaction Problems (CSP)
  - Finite domain variables
  - Fixed number of constraints over these variables

Context

- (Discrete) Combinatorial Problems
- NP-Complete/NP-Hard Problems
- Constraint Satisfaction Problems (CSP)
  - Finite domain variables
  - Fixed number of constraints over these variables
  - Is there a solution satisfying these constraints ?

Context

- (Discrete) Combinatorial Problems
- NP-Complete/NP-Hard Problems
- Constraint Satisfaction Problems (CSP)
  - Finite domain variables
  - Fixed number of constraints over these variables
  - Is there a solution satisfying these constraints ?
  - Is there a solution satisfying these constraints and optimizing a cost function f?

Context

- (Discrete) Combinatorial Problems
- NP-Complete/NP-Hard Problems
- Constraint Satisfaction Problems (CSP)
  - Finite domain variables
  - Fixed number of constraints over these variables
  - Is there a solution satisfying these constraints ?
  - Is there a solution satisfying these constraints and optimizing a cost function f?
- CP-Solvers : Branching + Propagation

What is a constraint?

A 'formal' definition

A constraint C defined on a set of variables $[X_1, X_2, .., X_n]$ defines a relation on the domains of $\mathcal{X}$.

Constraints can be given in

- Extension (i.e. table constraints) :

X_1	X_2	X_3	X_4
1	4	-8	0
-7	0	1	0
6	2	1	1
2	2	9	1

- Intention :
  - $X < Y$
  - $X \ mod\ 4 < |Y|$
  - $X \ mod\ 4 < Y \wedge |X| > 5 \wedge X \neq Y$

What is a constraint?

A 'formal' definition

A constraint C defined on a set of variables $[X_1, X_2, .., X_n]$ defines a relation on the domains of $\mathcal{X}$.

Constraints can be given in

- Extension (i.e. table constraints) :

X_1	X_2	X_3	X_4
1	4	-8	0
-7	0	1	0
6	2	1	1
2	2	9	1

- Intention :
  - $X < Y$
  - $X \ mod \ 4 < |Y|$
  - $X \ mod \ 4 < Y \wedge |X| > 5 \wedge X \neq Y$
- Each constraint is associated to a propagator
- A Constraint can be seen as a (sub)-problem

Arc Consistency

Definition

A constraint C is *Arc Consistent* (AC) iff for every variable x in its scope and every value v in the current domain of x, we can extend the current assignment to be consistent while assigning v to x.

Arc Consistency

Definition

A constraint C is *Arc Consistent* (AC) iff for every variable x in its scope and every value v in the current domain of x, we can extend the current assignment to be consistent while assigning v to x.

If C_1 is AC and C_2 is AC, does not imply $C_3 : C_1 \wedge C_2$ is AC!

Example

Let X_i be integer variables.

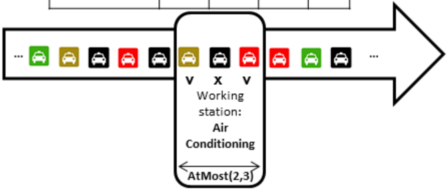
- $C_1 : \sum_{i=1}^{i=n} X_i \leq k$: is polynomial
- $C_2 : \sum_{i=1}^{i=n} X_i \geq k$ is polynomial
- $C_3 : C_1 \wedge C_2 : \sum_{i=1}^{i=n} X_i = k$ is *NP-Hard*!

Sequencing Problems

Sequencing Constraints : enforce upper and/or lower bounds on all (some) sub-sequences of variables of a given length within a main sequence.

The car-sequencing problem

Air Conditioning	V	X	V	X
Sunroof	X	V	X	V
MP3 Device	V	V	X	X



Constraints

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

Crew-rostering

	Week 1							W 2	W 3	W 4	d
emp_1	- - -	- - -	- - -	- - -	- - -	- - -	- - -				17
emp_2	- - -	- - -	- - -	- - -	- - -	- - -	- - -	..	..	..	17
..	- - -	- - -	- - -	- - -	- - -	- - -	- - -	..	..	..	17
emp_{20}	- - -	- - -	- - -	- - -	- - -	- - -	- - -	..	..	..	17
demande:	6;6;3	6;6;3	6;6;3	6;6;3	6;6;3	2;2;1	2;2;1	..	..	..	17*20

Constraints

- A required demand for each period.
- Each employee has to work 34 hours per week (17 shifts overall).
- Atmost 8h working shift per day.
- Atmost 5 days per each each 7 days period.

Definition

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

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

Definition

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

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

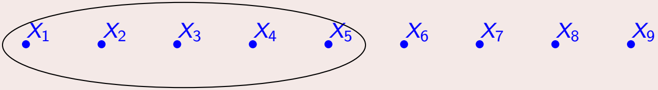
Example $\textsc{AtMostSeqCard}(2, 5, 4, [x_1, \ldots, x_9])$

Definition

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

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

Example AtMostSeqCard$(2, 5, 4, [x_1, \ldots, x_9])$

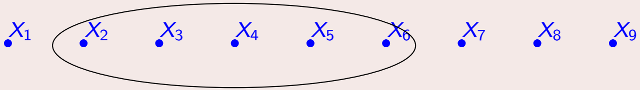


Definition

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

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

Example $\textsc{AtMostSeqCard}(2, 5, 4, [x_1, \ldots, x_9])$

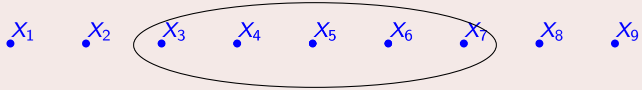


Definition

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

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

Example $\textsc{AtMostSeqCard}(2, 5, 4, [x_1, \ldots, x_9])$

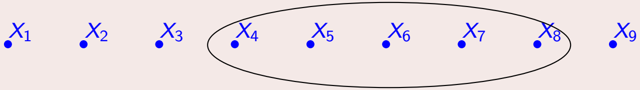


Definition

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

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

Example $\textsc{AtMostSeqCard}(2, 5, 4, [x_1, \ldots, x_9])$

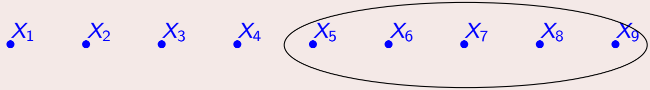


Definition

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

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

Example $\textsc{AtMostSeqCard}(2, 5, 4, [x_1, \dots, x_9])$

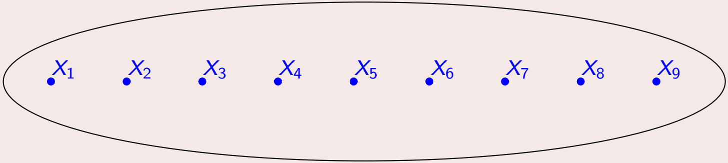


Definition

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

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

Example $\text{ATMOSTSEQCARD}(2, 5, 4, [x_1, \ldots, x_9])$

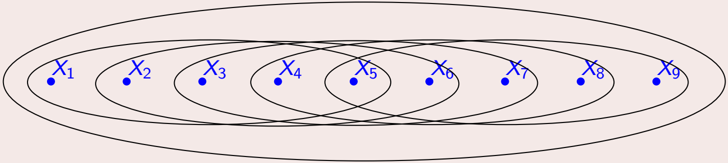


Definition

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

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

Example $\textsc{AtMostSeqCard}(2, 5, 4, [x_1, \ldots, x_9])$



Existing complexities

Gen-Sequence

- cost-Regular encoding: $O(2^q n)$ [Van Hoeve et al, 2009]
- Gen-Sequence: $O(n^3)$ [Van Hoeve et al, 2009]
- Flow-based Algorithm: $O(n^2)$ [Maher et al, 2008]

Gsc

- Gcc encoding, Does not achieve AC, NP-Hard [Puget and Régin, 1997]

The propagator

- `leftmost`: computes an assignment w maximizing the cardinality of the sequence with respect to the AtMost constraints.
- $Left[i] = \sum_{j=1}^{j=i} \texttt{leftmost}[j]$.
- $Right[i]$: same as $Left$ but in the reverse sense, i.e. $[x_n, .., x_1]$.

The propagator

- `leftmost`: computes an assignment w maximizing the cardinality of the sequence with respect to the AtMost constraints.
- $Left[i] = \sum_{j=1}^{j=i}$ `leftmost`$[j]$.
- $Right[i]$: same as *Left* but in the reverse sense, i.e. $[x_n, .., x_1]$.

AtMostSeqCard$(u = 4, q = 8, d = 12)$

$\mathcal{D}(x_i)$		.	0	.	.	.	.	.	.	0	1	0		.	.	.	.	.	.	.	.	.	1	
`leftmost`$[i]$		**1**	0	**1**	**1**	**1**	0	0	0	0	1	0	**1**	**1**	**1**	0	0	0	**1**	0	**1**	**1**	1	
$Left[i]$	0	1	1	2	3	4	4	4	4	4	4	**4**	5	6	7	7	7	7	8	8	9	10	10	
$Right[i]$		10	9	9	9	8	7	6	6	6	6	6	6	**5**	4	3	3	3	3	3	2	1	0	0

The propagator

- `leftmost`: computes an assignment w maximizing the cardinality of the sequence with respect to the AtMost constraints.
- $Left[i] = \sum_{j=1}^{j=i} \texttt{leftmost}[j]$.
- $Right[i]$: same as $Left$ but in the reverse sense, i.e. $[x_n, .., x_1]$.

AtMostSeqCard$(u = 4, q = 8, d = 12)$

$\mathcal{D}(x_i)$		.	0	.	.	.	.	.	.	0	1	0		.	.	.	.	.	.	.	.	.	1	
`leftmost`$[i]$		**1**	0	**1**	**1**	**1**	0	0	0	0	1	0	**1**	**1**	**1**	0	0	0	**1**	0	**1**	**1**	1	
$Left[i]$	0	1	1	2	3	4	4	4	4	4	4	**4**	5	6	7	7	7	7	8	8	9	10	10	
$Right[i]$		10	9	9	9	8	7	6	6	6	6	6	6	**5**	4	3	3	3	3	3	2	1	0	0

The propagator

- `leftmost`: computes an assignment w maximizing the cardinality of the sequence with respect to the AtMost constraints.
- $Left[i] = \sum_{j=1}^{j=i}$ `leftmost`$[j]$.
- $Right[i]$: same as *Left* but in the reverse sense, i.e. $[x_n, .., x_1]$.

AtMostSeqCard$(u = 4, q = 8, d = 12)$

$\mathcal{D}(x_i)$		.	0	.	.	.	.	.	.	0	1	0	**1**	.	.	.	.	.	.	.	.	.	1	
`leftmost`$[i]$		**1**	0	**1**	**1**	**1**	0	0	0	0	1	0	**1**	**1**	**1**	0	0	0	**1**	0	**1**	**1**	1	
$Left[i]$	0	1	1	2	3	4	4	4	4	4	4	**4**	5	6	7	7	7	7	8	8	9	10	10	
$Right[i]$		10	9	9	9	8	7	6	6	6	6	6	6	**5**	4	3	3	3	3	3	2	1	0	0

The propagator

- `leftmost`: computes an assignment w maximizing the cardinality of the sequence with respect to the AtMost constraints.
- $Left[i] = \sum_{j=1}^{j=i} \texttt{leftmost}[j]$.
- $Right[i]$: same as *Left* but in the reverse sense, i.e. $[x_n, .., x_1]$.

AtMostSeqCard$(u = 4, q = 8, d = 12)$

$\mathcal{D}(x_i)$		.	0	.	.	.	.	.	.	0	1	0		.	.	.	.	.	.	.	.	.	1	
`leftmost`$[i]$		**1**	0	**1**	**1**	**1**	0	0	0	0	1	0	**1**	**1**	**1**	0	0	0	**1**	0	**1**	**1**	1	
$Left[i]$	0	1	1	2	3	4	4	4	4	4	4	**4**	5	6	7	7	7	7	8	8	9	10	10	
$Right[i]$		10	9	9	9	8	7	6	6	6	6	6	6	**5**	4	3	3	3	3	3	2	1	0	0
GAC$(\mathcal{D}(x_i))$		**1**	0	.	.	.	.	**0**	**0**	0	1	0	**1**	**1**	**1**	**0**	**0**	**0**	.	.	**1**	**1**	1	

Arc consistency

- AC on each AtMost: $(\sum_{l=1}^{q} x_{i+l} \leq u)$
- AC on $\sum_{i=1}^{n} x_i = d$
- If $Left[n] < d$ Then *fail*
- If $Left[n] = d$ and $Left[i] + Right[n-i+1] \leq d$ Then $\mathcal{D}(x_i) \leftarrow \{0\}$
- If $Left[n] = d$ and $Left[i-1] + Right[n-i] < d$ Then $\mathcal{D}(x_i) \leftarrow \{1\}$

Extension

Extension

What about multiple AtMostSeqCard within the same sequence ?

Definition

$\textsc{MultiAtMostSeqCard}(u_1, .., u_m, q_1, .., q_m, d, [x_1, \ldots, x_n]) \Leftrightarrow$

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

Arc consistency

- MultiAtMostSeqCard can be modeled as a conjunction of AtMostSeqCard

Arc consistency

- MultiAtMostSeqCard can be modeled as a conjunction of AtMostSeqCard
- BUT This conjunction hinders propagation!

Arc consistency

- MultiAtMostSeqCard can be modeled as a conjunction of AtMostSeqCard
- BUT This conjunction hinders propagation!
- Fortunately We were able to find a way to extend the filtering algorithm to handle several AtMost constraints together.
- The complexity of achieving Arc Consistency is $O(m.n)$!

Car-Sequencing results

Table : Evaluation of the filtering methods (averaged over the 42 heuristics X 5 runs)

propagation	*set1* (70 × 5)			*set2* (4 × 5)			*set3* (5 × 5)			*set4* (7 × 5)		
	#sol	avg bts	time	#sol	avg bts	time	#sol	avg bts	time	#sol	avg bts	time
sum	11270	174017	10.49	124	1101723	58.75	0	-	$>$ 1200	99	378475	30.83
gsc	14008	1408	3.16	425	131062	109.45	31	55365	276.06	195	23897	53.61
amsc	13497	33600	3.79	**470**	**665205**	**70.56**	16	40326	8.62	**214**	**215349**	**38.45**
gsc+amsc	**14033**	**1007**	**3.03**	439	104823	99.71	**32**	**57725**	**285.43**	202	22974	61.61

Car-Sequencing results

Table : Optimization results

Instances	*amsc*			*gsc*			*gsc+amsc*			*sum*		
	Empty slots		time (s)	Empty slots		time (s)	Empty slots		time (s)	Empty slots		time (s)
	min	avg	avg	min	avg	avg	min	avg	avg	min	avg	avg
pb_200(..)	7.75	8.32	13.06	7.87	8.35	44.03	**7.62**	**8.27**	53.09	7.75	8.32	21.52
pb_300(..)	11.62	**12.37**	53.04	11.87	12.77	99.19	**11.50**	12.47	129.04	11.87	12.57	42.49
pb_400(..)	**10.57**	11.45	10.28	11.14	11.74	185.44	11.00	11.71	175.28	**10.57**	**11.34**	6.58

Crew-Rostering results

Table : Evaluation of the filtering methods

Heuristic	Most constrained employee						Most constrained shift					
Model	satifisable (1140)			unsatisfiable (385)			satifisable (1140)			unsatisfiable (385)		
	#sol	time	avg bts	#sol	time	avg bts	#sol	time	avg bts	#sol	time	avg bts
sum	772	21.93	205087	165	0.06	0	987	20.76	169964	352	19.74	180161
gsc	746	65.75	14133	175	0.98	0	1006	33.30	8875	335	15.97	5145
amsc	818	20.51	147479	215	0.13	330	1061	10.07	90247	362	12.19	108797
mamsc	**842**	**20.78**	**125886**	**270**	**0.05**	**0**	**1074**	**10.94**	**91222**	**377**	**14.63**	**110244**

SAT Solving

SAT

- A SAT Problem can be seen as a particular case of CSP
- Boolean variables
- 1 type of constraints : clauses (for instance $a \vee \neg b \vee \neg f \vee k$)

SAT Solving

SAT

- A SAT Problem can be seen as a particular case of CSP
- Boolean variables
- 1 type of constraints : clauses (for instance $a \vee \neg b \vee \neg f \vee k$)

The SAT revolution

- SAT is becoming a community!
- SAT Solvers are ever evolving
- SAT is being used applied to a wide range of combinatorial (optimization) problems (For instance : state-of-the-art results in RCPSP [Andreas Schutt et.al 2013])

Lazy Clause Generation

SAT & CP :

- Can we get the best from both approaches?
- to encode into SAT or to use global constraints?
  $\rightarrow$A key concept in hybrid solvers : Explanations

An explanation is a set atomic constraints triggering a failure/filtering.

example

Cardinality Constraint : $\sum_{i=1}^{n} x_i \leq k$; $D(x_i) = \{0, 1\}$.
$x_i \leftarrow 1$ is pruned if we already have k appearances of the value 1.

$$\{x_j \leftarrow 1 | D(x_j) = \{1\}\} \ \rightarrow x_i \not\leftarrow 1 \ .$$

Lazy Clause Generation

SAT & CP :

- Can we get the best from both approaches?
- to encode into SAT or to use global constraints?
  $\rightarrow$A key concept in hybrid solvers : Explanations

An explanation is a set atomic constraints triggering a failure/filtering.

example

Cardinality Constraint : $\sum_{i=1}^{n} x_i \leq k$; $D(x_i) = \{0, 1\}$.
$x_i \leftarrow 1$ is pruned if we already have k appearances of the value 1.

$$\{x_j \leftarrow 1 | D(x_j) = \{1\}\} \quad \rightarrow x_i \not\leftarrow 1 \ .$$

Explaining AtMostSeqCard : the key idea

Explaining Failure

1. If a failure is triggered by a cardinality constraint (i.e. ($\sum_{l=1}^{q} x_{i+l} \leq u$) or $\sum_{i=1}^{n} x_i = d$), then it is easy to generate an explanation.
2. If a failure triggered by $Left[n] < d$, a naive explanation would be the set of all assignments in the sequence.

Some observations

Let $max(i)$ be the maximum cardinality of the q subsequences involving x_i when computing `leftmost`[i].

Some observations

Let $max(i)$ be the maximum cardinality of the q subsequences involving x_i when computing `leftmost`[i].

Let S : 1 1 0 0 . subject to AtMost(2/5).
→`leftmost` on S gives 1 1 0 0 0

Consider the sequence S_0 : 1 1 . 0 .
→`leftmost` on S_0 gives 1 1 0 0 0

Some observations

Let $max(i)$ be the maximum cardinality of the q subsequences involving x_i when computing `leftmost`[i].

Let S : 1 1 0 0 . subject to AtMost(2/5).
→`leftmost` on S gives 1 1 0 0 0

Consider the sequence S_0 : 1 1 . 0 .
→`leftmost` on S_0 gives 1 1 0 0 0

$$\{x_i \leftarrow 0 \mid max(i) = u\}$$

Some observations

Let $max(i)$ be the maximum cardinality of the q subsequences involving x_i when computing `leftmost`[i].

Let S : 1 1 0 0 . subject to AtMost(2/5).
→`leftmost` on S gives 1 1 0 0 0

Consider the sequence S_0 : 1 1 . 0 .
→`leftmost` on S_0 gives 1 1 0 0 0

$$\{x_i \leftarrow 0 \mid max(i) = u\}$$

Consider the sequence S_2 : . 1 0 0 .
→`leftmost` on S_2 gives 1 1 0 0 0

Some observations

Let $max(i)$ be the maximum cardinality of the q subsequences involving x_i when computing `leftmost`[i].

Let S : 1 1 0 0 . subject to AtMost(2/5).
→`leftmost` on S gives 1 1 0 0 0

Consider the sequence S_0 : 1 1 . 0 .
→`leftmost` on S_0 gives 1 1 0 0 0

$$\{x_i \leftarrow 0 \mid max(i) = u\}$$

Consider the sequence S_2 : . 1 0 0 .
→`leftmost` on S_2 gives 1 1 0 0 0

$$\{x_i \leftarrow 1 \mid max(i) \neq u\}$$

Theorem

Theorem

Let S be the set of all assignments,
$S^* = S \setminus (\{x_i \leftarrow 0 \mid max(i) = u\} \cup \{x_i \leftarrow 1 \mid max(i) \neq u\})$, *then*
S^* *is a valid explanation.*

→runs in $O(n)$ since we call `leftmost` once.

Example : AtMostSeqCard$(2, 5, 8, [x_1, ..x_{22}])$

S	1	0	1	0	0	.	.	0	0	0	1	1	0	0	0	0	1	0	0	0	0	1
leftmost$(S(x_i))$	1	0	1	0	0	1	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	1
$Left[i]$	1	1	2	2	2	3	3	3	3	3	4	5	5	5	5	5	6	6	6	6	6	7

$$Left[22] = 7 < 8 : \text{ FAILURE}$$

$max(i)$	2	2	2	2	2	1	2	2	2	2	2	2	2	2	2	1	1	1	1	1	1	1
S^*	1	.	1	.	.	.	.	.	.	.	1	1	.	.	.	0	.	0	0	0	0	.

The final explanation size $|S^*|$ is 9 while the naive one ($|S|$) is 20.

Explaining pruning

explanation for $x \leftarrow k$?

1. Add $x \not\leftarrow k$ to the instantiation where the pruning was performed.
2. Use the previous procedure to explain the failure on the new instantiation.

Experimental results

Table : Evaluation of the models

Method	sat[easy] (74 × 5)			sat[hard] (7 × 5)			unsat/unknown (28 × 5)		
	#suc	avg fails	time	#suc	avg fails	time	#suc	avg fails	time
Hybrid (VSIDS)	370	903	0.23	16	207211	286.32	35	177806	224.78
Hybrid (VSIDS → Slot)	370	739	0.23	35	76256	64.52	37	204858	248.24
Hybrid (Slot → VSIDS)	370	132	0.04	34	4568	2.50	37	234800	287.61
Hybrid (Slot)	370	132	0.04	**35**	**6304**	**3.75**	23	174097	299.24
CP	**370**	**43.06**	**0.03**	35	57966	16.25	0	-	-

Contributions & Future Research

Contributions

- AtMostSeqCard : best existing complexity: $O(n^2)$ [Maher et al, 2008].
- An Arc Consistency algorithm with an optimal worst case time complexity O(n).
  - Car-sequencing
  - Crew-Rostering
- Useful Extensions of the AtMostSeqCard constraint
- A linear time explanation for the AtMostSeqCard constraint
- NICTA collaboration : An Empirical study between CP, Hybrid CP/SAT and pure SAT Models for the Car-Sequencing problem
  - Closing 13 out of the 23 large open instances.

Perspectives

- How hard is finding optimal explanations for AtMostSeqCard ? (it is NP-Hard in general)
- How to explain MultiAtMostSeqCard ?
- MultiAtMostSeqCard + explanation for other Timetabling problems?

Thank you!

Related publications

Journals

1 Mohamed Siala, Emmanuel Hebrard and Marie-Jose Huguet, *An Optimal Arc Consistency Algorithm for a Particular Case of Sequence Constraint*, **Constraints** January 2014, Volume 19, Issue 1, pp 30-56

International Conferences

2- **[Honorable mention]** Mohamed Siala, Emmanuel Hebrard, and Marie-Josè Huguet, *An Optimal Arc Consistency Algorithm for a Chain of Atmost Constraints with Cardinality*,**CP 12**, Quebec, Canada

3-Christian Artigues, Emmanuel Hebrard, Valentin Mayer-Eichberger*, Mohamed Siala*, and Toby Walsh, *SAT and Hybrid Models of the Car-Sequencing problem*, **CP-AI-OR 14**, Cork, Ireland.

Workshops

4-Mohamed Siala, Emmanuel Hebrard, and Marie-José Huguet, *A Study of Branching Heuristics for the Car-Sequencing Problem* , SSNOW'12, CPAIOR'12, Nantes, France

5-Mohamed Siala, Christian Artigues, Emmanuel Hebrard, *Explaining the AtMostSeqCard constraint*, CP'13 Doctoral Program, September 2013, Uppsala, Sweden.