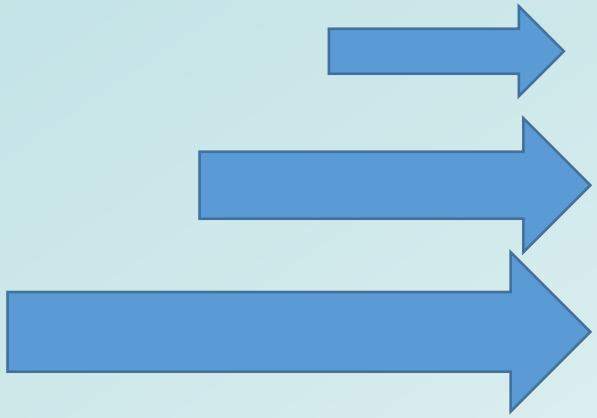
Efficient Constraint Programming Approaches for routing problem: a case study for the VRP

Matthieu Gondran
Eric Bourreau
and Philippe Lacomme

Motivation

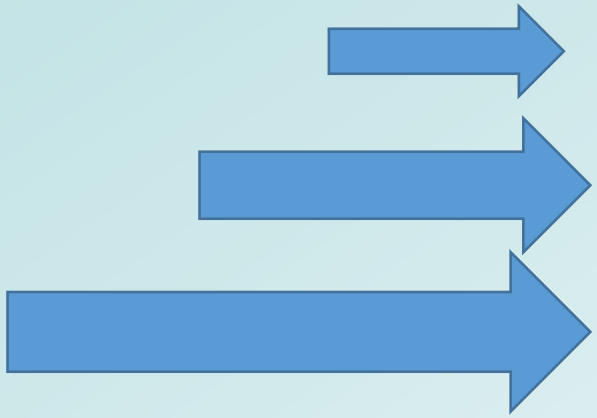
- Constraint Programming (CP)
  - Starts in early 80's
  - Originally from Artificial Intelligence
  - Successfully applied to combinatorial problems around 2000.
- Few works around routing problems
  - Most of (nice) papers FROM the constraint community TO the constraint community
- The GAP from ILP to CP is not too big
  - Model & Solve (declarative but better if you manage the search strategy)
  - Very (very) efficient free/commercial solvers
  - Easy hybridization

VRP: linear formulation



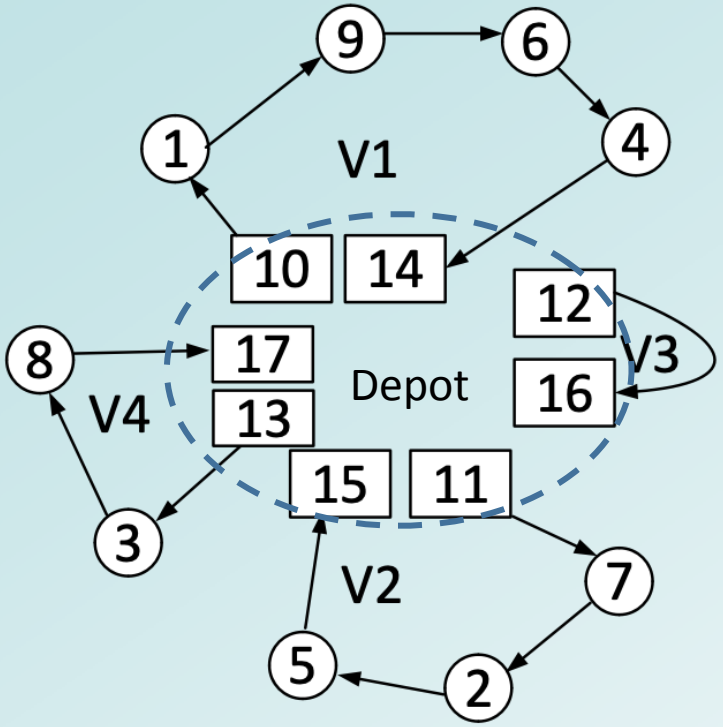
$\forall k = 1..V$	$\sum_{j=2}^{N} x_{1j}^{k} \leq 1$	(1)
$\forall i = 1..N, \forall j = 1..N, \forall k = 1..V$	$q_i^k + D_j \leq q_j^k + \left(1 - x_{1j}^k\right).H$	(2)
$\forall j = 1..N\ , \forall k = 1..V$	$q_j^k \leq C$	(3)
$\forall i = 2..N$	$\sum_{j=1}^{N} \sum_{k=1}^{K} x_{ij}^{k} = 1$	(4)
$\forall j = 2..N$	$\sum_{i=1}^{N} \sum_{k=1}^{K} x_{ij}^{k} = 1$	(5)
$\forall j = 2..N, \forall k = 1..V$	$\sum_{j=1}^{N} x_{ji}^{k} = \sum_{j=1}^{N} x_{ij}^{k}$	(6)
$\forall i = 1..N, \forall k = 1..V$	$x_{ji}^{k} \neq 1$	(7)
	$d = \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{K} x_{ij}^{k}.T_{ij}$	(8)
	$Min\ d$	

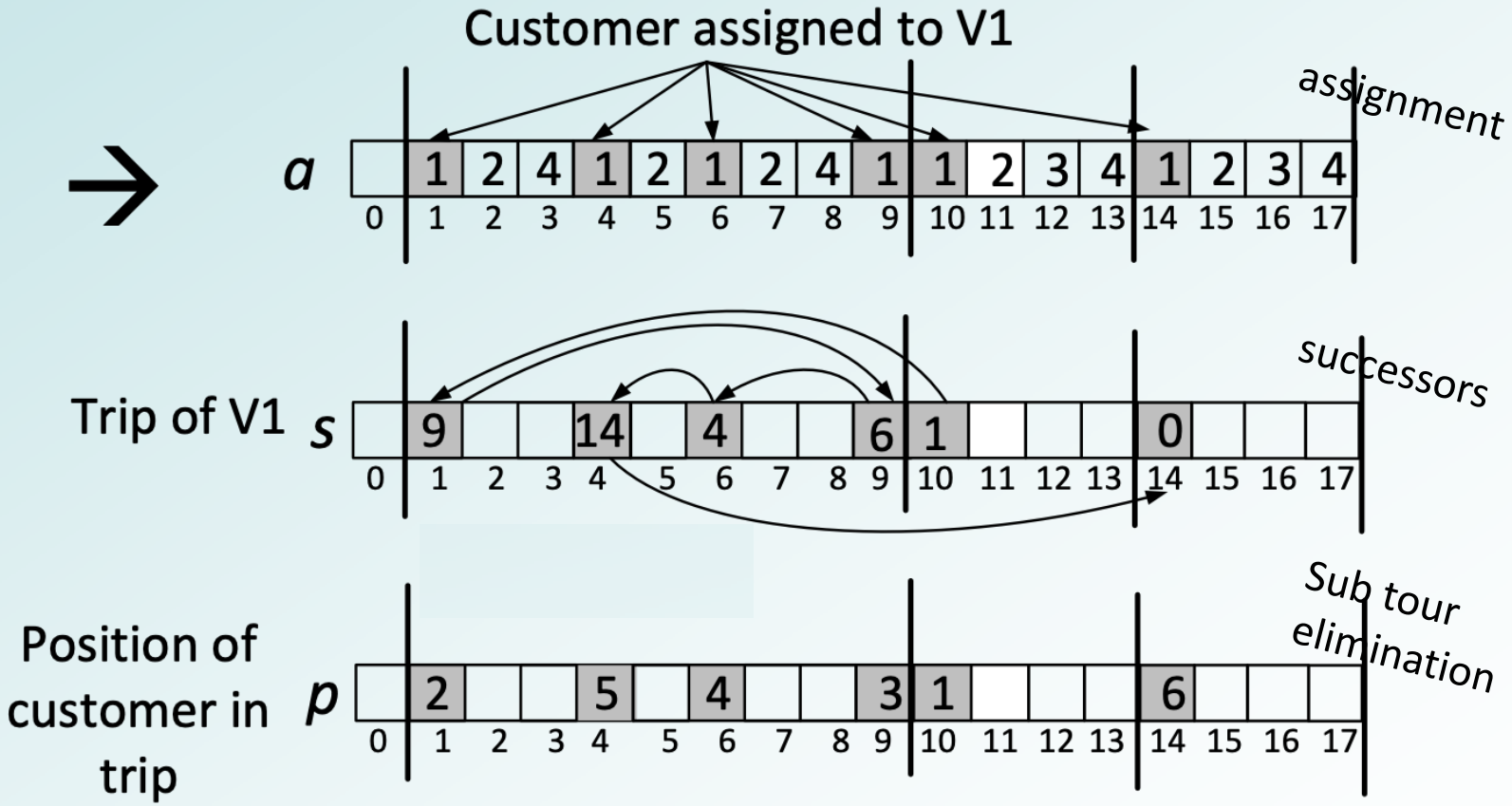
VRP: linear formulation



$\forall k = 1..V$	$\sum_{j=2}^{N} x_{1j}^{k} \leq 1$	(1)	assignment
$\forall i = 1..N, \forall j = 1..N, \forall k = 1..V$	$q_i^k + D_j \leq q_j^k + (1 - x_{1j}^k).H$	(2)	Sub tour elimination
$\forall j = 1..N, \forall k = 1..V$	$q_j^k \leq C$	(3)	
$\forall i = 2..N$	$\sum_{j=1}^{N} \sum_{k=1}^{K} x_{ij}^{k} = 1$	(4)	predecessors successors
$\forall j = 2..N$	$\sum_{i=1}^{N} \sum_{k=1}^{K} x_{ij}^{k} = 1$	(5)	
$\forall j = 2..N, \forall k = 1..V$	$\sum_{j=1}^{N} x_{ji}^{k} = \sum_{j=1}^{N} x_{ij}^{k}$	(6)	
$\forall i = 1..N, \forall k = 1..V$	$x_{ji}^{k} \neq 1$	(7)	
	$d = \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{K} x_{ij}^{k}.T_{ij}$	(8)	
	$Min\ d$		

VRP: CP formulation





VRP: CP formulation

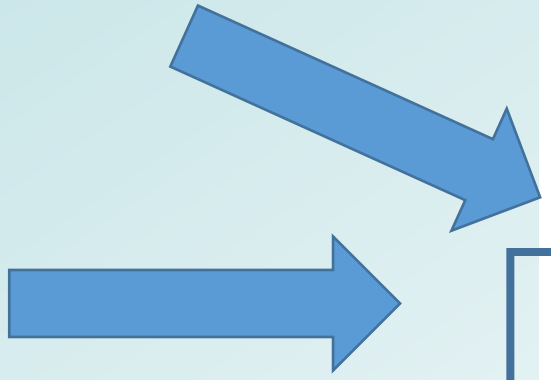
Data

N	Set of customer to service *with* |N| the number of customers
V	Set of vehicles
$S = \|N\| + 2\|V\|$	Number of nodes with 2|V| the total number of depots
V^d	Set of initial depot node (one per vehicle)
V^f	Set of final depot node (one per vehicle)
D_i	quantity to deliver at node i
T_{ij}	distance from i to j with $i \in N$ et $j \in N$
T'_{ij}	distance from i to j with $i \in N \cup V^d \cup V^f$ et $j \in N \cup V^d \cup V^f$
C_v	vehicle capacity $v \in V$

Variables

p_i	$= k$, the customer i is in position k in the trip
s_i	successor of i ($\forall i \in N \cup V^d$) in the trip
a_i	$= v$ assignment of vehicle v to service i ($\forall i \in N \cup V^d \cup V^f$)
d	total cost
b^v	set of customers assigned to vehicle v

VRP: CP formulation – mathematical formulation



$\forall i \in N \cup V^d$	$s_i \in [1; S]$	(1)
$\forall i \in V^f$	$s_i = 0$	(2)
$\forall i \in N$	$a_i \in [1; \lvert V \rvert]$	(3)
$\forall i \in V^f \cup V^d$	$a_i = i$	(4)
$\forall i \in N \cup V^f$	$p_i \in [1; \lvert N \rvert + 1]$	(5)
$\forall i \in V^d$	$p_i = 0$	(6)
$\forall i, j \in \{N \cup V^d \cup V^f\}^2$	$s_i \neq s_j$	(7)
$\forall i \in N \cup V^d$	$a_i = a_{s_i}$	(8)
$\forall i \in N \cup V^d$	$s_i \neq i$	(9)
$\forall i \in N \cup V^d$	$p_{s_i} = p_i + 1$	(10)
$\forall v \in V$,	$b^v = \{u \in N / a_u = v\}$	(11)
$\forall v \in V$,	$\sum_{i \in b^v} D_i \leq C_v$	(12)
	$d = \sum_{i=1}^{S} T_{i,s_i}$	(13)

VRP: CP formulation

$\forall i \in N \cup V^d$	$s_i \in [1; S]$	(1)
$\forall i \in V^f$	$s_i = 0$	(2)
$\forall i \in N$	$a_i \in [1; \|V\|]$	(3)
$\forall i \in V^f \cup V^d$	$a_i = i$	(4)
$\forall i \in N \cup V^f$	$p_i \in [1; \|N\| + 1]$	(5)
$\forall i \in V^d$	$p_i = 0$	(6)
$\forall i, j \in \{N \cup V^d \cup V^f\}^2$	$AllDifferent(s_i)$	(7)
$\forall i \in N \cup V^d$	$Element(a_i, a, s_i)$	(8)
$\forall i \in N \cup V^d$	$s_i \neq i$	(9)
$\forall i \in N \cup V^d$	$t_i = p_i + 1$	(10.1)
$\forall i \in N \cup V^d$	$Element(t_i, p, s_i)$	(10.2)
$\forall v \in V$	$b^v = \{u \in N / a_u = v\}$	(11.1)
$\forall v \in V$	$setsIntsChannealing(b, a)$	(11.2)
$\forall v \in V$	$SumElements(b_i, D, C_v)$	(12)
$\forall i \in N \cup V^d \cup V^f$	$dp_i \in [1; H]$	(13.1)
$\forall i \in N \cup V^d \cup V^f$	$Element(dp_i, T_i, s_i)$	(13.2)
	$d = sum(dp_i)$	(13.3)

VRP: CP formulation

Element(X,T,I) ↔ T[I]=X

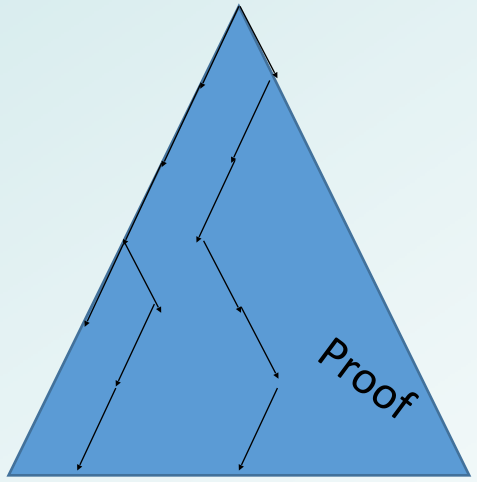
$\forall i \in N \cup V^d$	$s_i \in [1; S]$	(1)
$\forall i \in V^f$	$s_i = 0$	(2)
$\forall i \in N$	$a_i \in [1; \|V\|]$	(3)
$\forall i \in V^f \cup V^d$	$a_i = i$	(4)
$\forall i \in N \cup V^f$	$p_i \in [1; \|N\| + 1]$	(5)
$\forall i \in V^d$	$p_i = 0$	(6)
$\forall i, j \in \{N \cup V^d \cup V^f\}^2$	$AllDifferent(s_i)$	(7)
$\forall i \in N \cup V^d$	$Element(a_i, a, s_i)$	(8)
$\forall i \in N \cup V^d$	$s_i \neq i$	(9)
$\forall i \in N \cup V^d$	$t_i = p_i + 1$	(10.1)
$\forall i \in N \cup V^d$	$Element(t_i, p, s_i)$	(10.2)
$\forall v \in V$	$b^v = \{u \in N / a_u = v\}$	(11.1)
$\forall v \in V$	$setsIntsChannealing(b, a)$	(11.2)
$\forall v \in V$	$SumElements(b_i, D, C_v)$	(12)
$\forall i \in N \cup V^d \cup V^f$	$dp_i \in [1; H]$	(13.1)
$\forall i \in N \cup V^d \cup V^f$	$Element(dp_i, T_i, s_i)$	(13.2)
	$d = sum(dp_i)$	(13.3)

Keypoints

1. **Symmetry** breaks
2. **Link variables** to avoid iterative explicit enumeration of several trees
3. Promote **global constraints** including cumulative() and diffN() strongly powerful for scheduling and routing
4. **Avoid** « bad » constraint including sum() / scalar() with **poor propagation**
5. **Redundant constraint** can favor propagation
   - Graph : include both successor and predecessor (see Channeling)
6. **Search strategies**
   - Promote Variable selector considering first **maxRegret(),** which chooses the variable with the largest difference between the two smallest values in its domain
   - Assignment strategy → consider first **DomOverWDeg()**

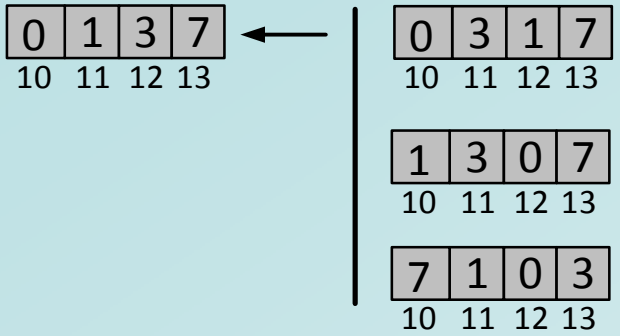
Modelling improvements

		Version V1
1 SOLUTION	S*	32
	Nb Nodes	29
	Backtracks	3
	Fails	3
	T*(sec)	0.023
OPTIMAL SOLUTION	S*	25
	Nb Nodes	51 992
	Backtracks	103 921
	Fails	51 960
	T*(sec)	1.936
PROOF	Nb Nodes	4 817 732
	Backtracks	9 635 453
	Fails	4 817 721
	TT(sec)	≅150.000



First sol ... Optimal sol

1/6 Model with symmetries breaks



$$\forall i \in V^d \qquad s_i < s_{i+1} \qquad (14)$$

		Version V1	Version V2
SYMETRIE			×
1 SOLUTION	S*	32	27
	Nb Nodes	29	28
	Backtracks	3	1
	Fails	3	1
	T*(sec)	0.023	0.026
OPTIMAL SOLUTION	S*	25	25
	Nb Nodes	51 992	2 847
	Backtracks	103 921	5 642
	Fails	51 960	2 823
	T*(sec)	1.936	0.346
PROOF	Nb Nodes	4 817 732	327 391
	Backtracks	9 635 453	654 777
	Fails	4 817 721	327 386
	TT(sec)	148.450	9.395

2/6 Link variables

$$\forall i \in V^d \qquad y_i = 100 \times a_i + s_i \tag{15}$$

$$\forall i \in N \qquad y_i \in [1; 100 \times |V| + S] \tag{16}$$

		Version V1	Version V2	Version V3
SYMETRIE			×	×
LING a AND s				×
1 SOLUTION	S*	32	27	27
	Nb Nodes	29	28	30
	Backtracks	3	1	1
	Fails	3	1	1
	T*(sec)	0.023	0.026	0.039
OPTIMAL SOLUTION	S*	25	25	25
	Nb Nodes	51 992	2 847	3 156
	Backtracks	103 921	5 642	6 252
	Fails	51 960	2 823	3 128
	T*(sec)	1,936	0,346	0.440
PROOF	Nb Nodes	4 817 732	327 391	188 107
	Backtracks	9 635 453	654 777	376 409
	Fails	4 817 721	327 386	188 202
	TT(sec)	148.450	9.395	7.154

Branch on decision variables ➔ y

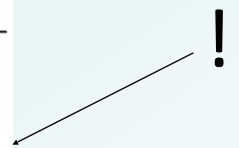
```
mon_solveur.setSearch( new DomOverWDeg(y, 0, new IntDomainMin()));
```

		Version V1	Version V2	Version V3	Version V4
SYMETRIE			×	×	×
LINK a AND s				×	×
BRANCH Y					×
1 SOLUTION	S*	32	27	27	37
	Nb Nodes	29	28	30	7
	Backtracks	3	1	1	0
	Fails	3	1	1	0
	T*(sec)	0.023	0.026	0.039	0.039
OPTIMAL SOLUTION	S*	25	25	25	25
	Nb Nodes	51 992	2 847	3 156	17 529
	Backtracks	103 921	5 642	6 252	34 996
	Fails	51 960	2 823	3 128	17 496
	T*(sec)	1.936	0.346	0.440	1.352
PROOF	Nb Nodes	4 817 732	327 391	188 107	78 569
	Backtracks	9 635 453	654 777	376 409	157 113
	Fails	4 817 721	327 386	188 202	78 544
	TT(sec)	148.450	9.395	7.154	3.787

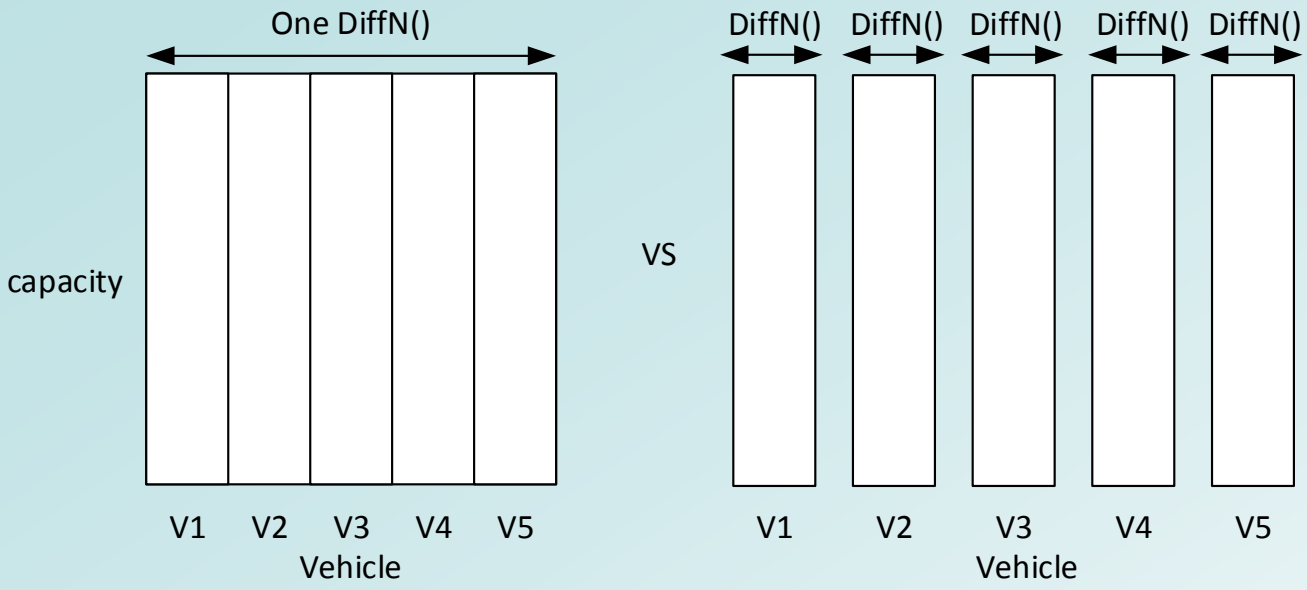
Branch on decision variables ➔ y

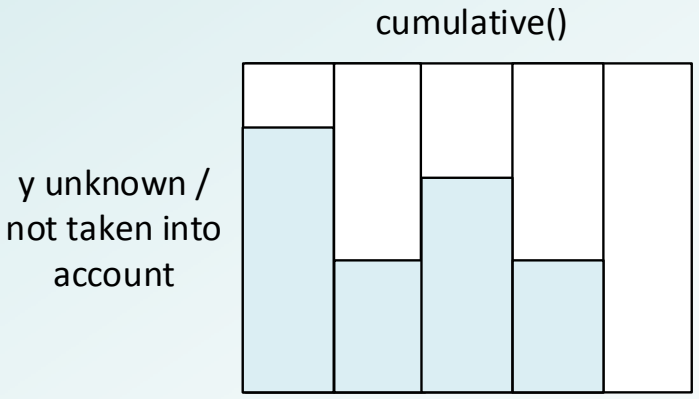
```
mon_solveur.setSearch( new DomOverWDeg(y, 0, new IntDomainMin()));
```

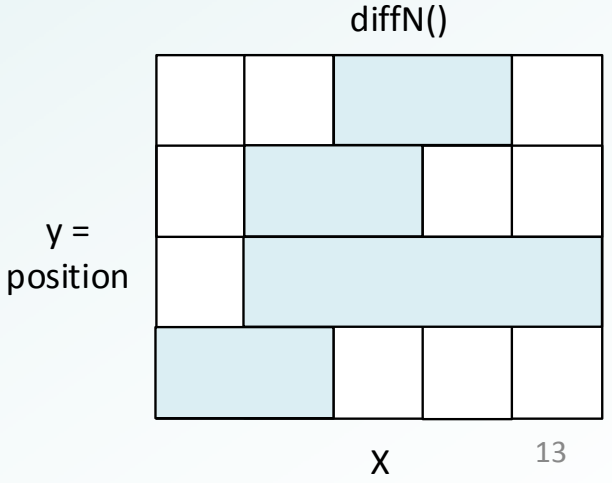
		Version V1	Version V2	Version V3	Version V4
SYMETRIE			×	×	×
LINK a AND s				×	×
BRANCH Y					×
1 SOLUTION	S*	32	27	27	37
	Nb Nodes	29	28	30	7
	Backtracks	3	1	1	0
	Fails	3	1	1	0
	T*(sec)	0.023	0.026	0.039	0.039
OPTIMAL SOLUTION	S*	25	25	25	25
	Nb Nodes	51 992	2 847	3 156	17 529
	Backtracks	103 921	5 642	6 252	34 996
	Fails	51 960	2 823	3 128	17 496
	T*(sec)	1.936	0.346	0.440	1.352
PROOF	Nb Nodes	4 817 732	327 391	188 107	78 569
	Backtracks	9 635 453	654 777	376 409	157 113
	Fails	4 817 721	327 386	188 202	78 544
	TT(sec)	148.450	9.395	7.154	3.787



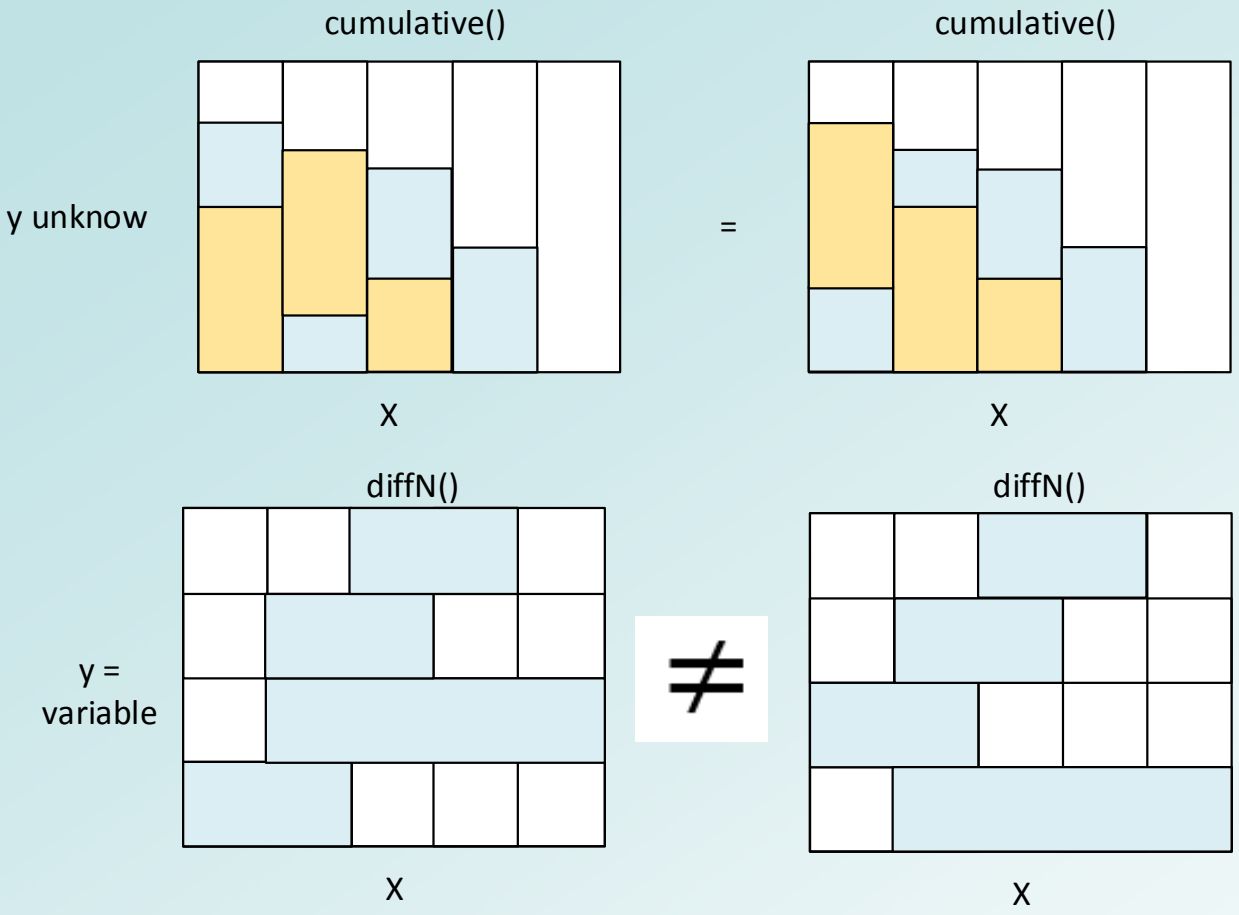
3/6 - Add global constraints

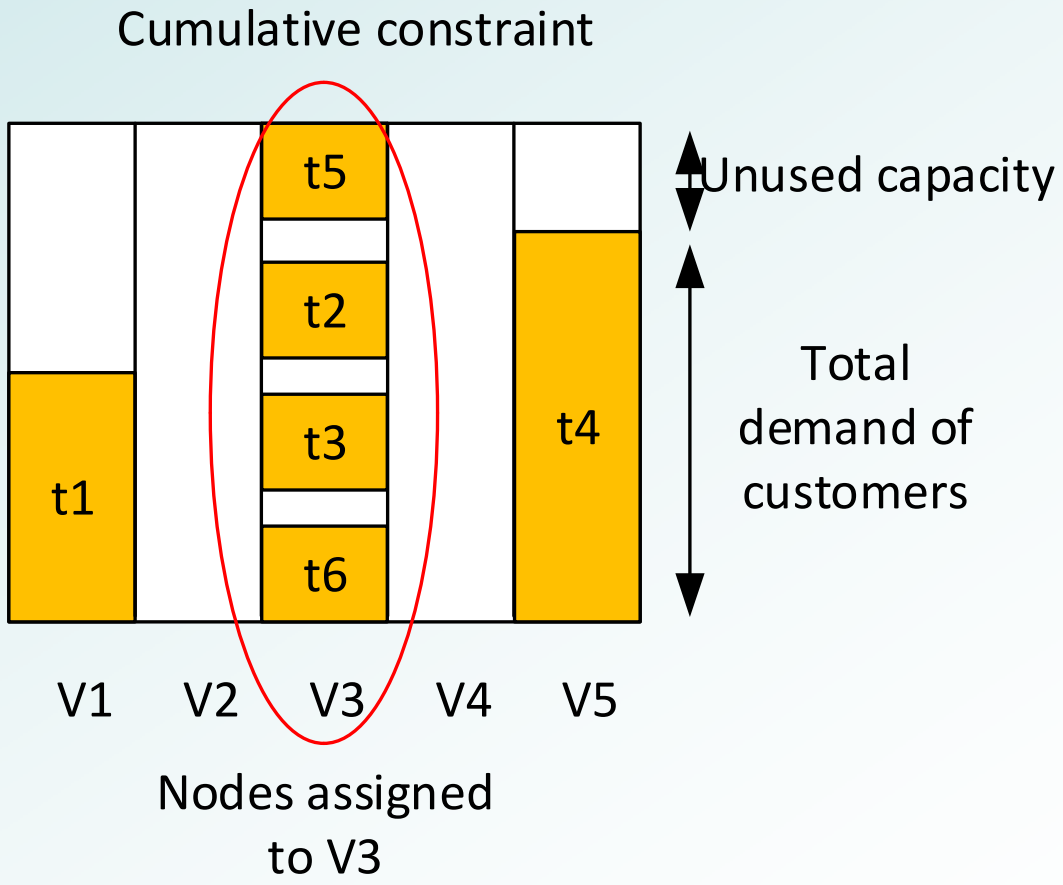






Add global constraints

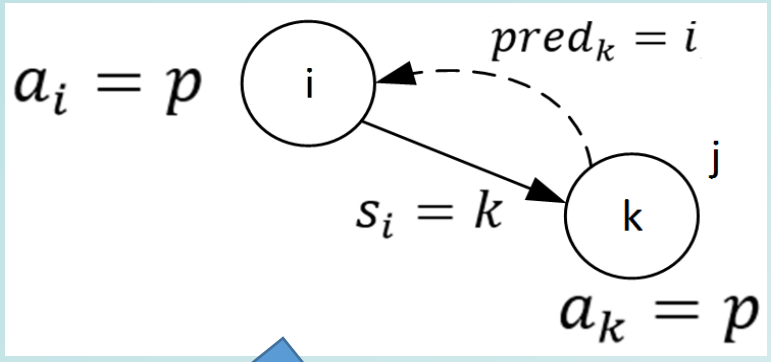




Add global constraints

		V1	V2	V3	V4	V5
SYMETRIE			×	×	×	×
LINK a AND s				×	×	×
BRANCH Y					×	×
CUMULATIVE					×	×
1 SOLUTION	S*	32	27	27	37	37
	Nb Nodes	29	28	30	7	7
	Backtracks	3	1	1	0	0
	Fails	3	1	1	0	0
	T*(sec)	0.023	0.026	0.039	0.039	0.036
OPTIMAL SOLUTION	S*	25	25	25	25	25
	Nb Nodes	51 992	2 847	3 156	17 529	17 529
	Backtracks	103 921	5 642	6 252	34 996	34 996
	Fails	51 960	2 823	3 128	17 496	17 496
	T*(sec)	1.936	0.346	0.440	1.352	1.591
PROOF	Nb Nodes	4 817 732	327 391	188 107	78 569	78 569
	Backtracks	9 635 453	654 777	376 409	157 113	157 113
	Fails	4 817 721	327 386	188 202	78 544	78 544
	TT(sec)	148.450	9.395	7.154	3.787	4.583

4/6 Channeling constraints



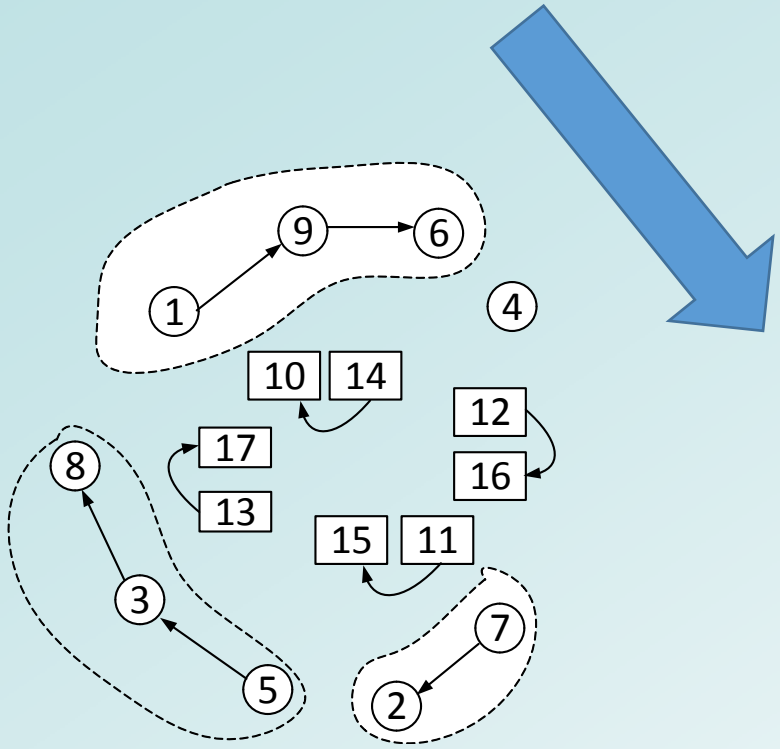
$\forall i \in V^d$	$s_i < s_{i+1}$	(14)
$\forall i \in V^d$	$y_i = 100 \times a_i + s_i$	(15)
$\forall i \in N$	$y_i \in [1; 100 \times \lvert V \rvert + S]$	(16)
$\forall i \in N \cup V^d$	$(pred_k = i) \Leftrightarrow (s_i = k)$	(17)
$\forall i \in N \cup V^d$	$pred_i \in [1; S]$	(18)
$\forall i \in V^f$	$pred_i = j$ avec V^d	(19)

Channeling constraints

		V1	V2	V3	V4	V5	V6
SYMETRIE			×	×	×	×	×
LINK a AND s				×	×	×	×
BRANCH Y					×	×	×
CUMULATIVE						×	×
CHANNELING							×
1 SOLUTION	S*	32	27	27	37	37	37
	Nb Nodes	29	28	30	7	7	7
	Backtracks	3	1	1	0	0	0
	Fails	3	1	1	0	0	0
	T*(sec)	0.023	0.026	0.039	0.039	0.036	0.046
OPTIMAL SOLUTION	S*	25	25	25	25	25	25
	Nb Nodes	51 992	2 847	3 156	17 529	17 529	17 529
	Backtracks	103 921	5 642	6 252	34 996	34 996	34 996
	Fails	51 960	2 823	3 128	17 496	17 496	17 496
	T*(sec)	1.936	0.346	0.440	1.352	1.591	1.966
PROOF	Nb Nodes	4 817	327	188	78 569	78 569	78 569
		732	391	107			
	Backtracks	9 635	654	376	157	157	157
		453	777	409	113	113	113
	Fails	4 817	327	188	78 544	78 544	78 544
		721	386	202			
	TT(sec)	148.450	9.395	7.154	3.787	4.583	5.740

5/6 - Check constraint during propagation

$$\forall v \in V, \quad \sum_{i \in b^v} D_i \leq C_v \qquad (12)$$



$\forall i \in V^d$	$s_i < s_{i+1}$	(14)
$\forall i \in V^d$	$y_i = 100 \times a_i + s_i$	(15)
$\forall i \in N$	$y_i \in [1; 100 \times \lvert V \rvert + S]$	(16)
$\forall i \in N \cup V^d$	$(pred_k = i) \Leftrightarrow (s_i = k)$	(17)
$\forall i \in N \cup V^d$	$pred_i \in [1; S]$	(18)
$\forall i \in V^f$	$pred_i = j$ avec V^d	(19)
$\forall i \in \{N \cup V^d \cup V^f\}$	$CPS_i \in [1; C_v]$	(20)
$\forall i \in \{N \cup V^d \cup V^f\}$	$CPS_{s_i} = CPS_i + D_i$	(21)

Check constraint during propagation

		V1	V2	V3	V4	V5	V6	V7
SYMETRIE			×	×	×	×	×	×
LINK a AND s				×	×	×	×	×
BRANCH Y					×	×	×	×
CUMULATIVE						×	×	×
CHANNELING							×	×
CAPASUM								×
1 SOLUTION	S*	32	27	27	37	37	37	37
	Nb Nodes	29	28	30	7	7	7	7
	Backtracks	3	1	1	0	0	0	0
	Fails	3	1	1	0	0	0	0
	T*(sec)	0.023	0.026	0.039	0.039	0.036	0.046	0.045
OPTIMAL SOLUTION	S*	25	25	25	25	25	25	25
	Nb Nodes	51 992	2 847	3 156	17 529	17 529	17 529	8 728
	Backtracks	103 921	5 642	6 252	34 996	34 996	34 996	17 404
	Fails	51 960	2 823	3 128	17 496	17 496	17 496	8 697
	T*(sec)	1.936	0.346	0.440	1.352	1.591	1.966	1.523
PROOF	Nb Nodes	4 817 732	327 391	188 107	78 569	78 569	78 569	38 879
	Backtracks	9 635 453	654 777	376 409	157 113	157 113	157 113	77 733
	Fails	4 817 721	327 386	188 202	78 544	78 544	78 544	38 854
	TT(sec)	148.450	9.395	7.154	3.787	4.583	5.740	4.263

6/6 - Branch on neighborhoods

$$T'_{ij} = T_{ij} \times 100 + j$$

$$DPS_i = T'_{i,s_i}$$

```
mon_solveur.setSearch(
    new DomOverWDeg(dps,
                    0,
                    new IntDomainMin()
                   )
) ;
```

```
mon_solveur.setSearch(
  Search.intVarSearch(
      new MaxRegret(),
      new IntDomainMin(),
      dps)
);
```

$\forall i \in V^d$	$s_i < s_{i+1}$	(14)
~~$\forall i \in V^d$~~	~~$y_i = 100 \times a_i + s_i$~~	~~(15)~~
~~$\forall i \in N$~~	~~$y_i \in [1; 100 \times \lvert V \rvert + S]$~~	~~(16)~~
$\forall i \in N \cup V^d$	$(pred_k = i) \Leftrightarrow (s_i = k)$	(17)
$\forall i \in N \cup V^d$	$pred_i \in [1; S]$	(18)
$\forall i \in V^f$	$pred_i = j$ avec V^d	(19)
$\forall i \in \{N \cup V^d \cup V^f\}$	$CPS_{s_i} = CPS_i + D_i$	(20)
$\forall i \in \{N \cup V^d \cup V^f\}$	$CPS_i \in [0; C_v]$	(21)
$\forall i \in \{N \cup V^d \cup V^f\}$	$T'_{ij} = T_{ij} \times 100 + j$	(22)
$\forall i \in \{N \cup V^d \cup V^f\}$	$DPS_i = T'_{i,s_i}$	(23)
$\forall i \in \{N \cup V^d \cup V^f\}$	$DPS_i \in [1; S]$	(24)

Modelling improvements: all advices together

		V1	V2	V3	V4	V5	V6	V7	V8	V9
SYMETRIE			×	×	×	×	×	×	×	×
LINK α AND s				×	×	×	×	×	×	×
BRANCH Y					×	×	×	×		
CUMULATIVE						×	×	×	×	×
CHANNELING							×	×	×	×
CAPASUM								×	×	×
DOMOVERWDEG (DPS)									×	
MAXREGRET (DPS)										×
1 SOLUTION	S*	32	27	27	37	37	37	37	26	25
	Nb Nœuds	29	28	30	7	7	7	7	21 367	24
	Backtracks	3	1	1	0	0	0	0	42 715	31
	Fails	3	1	1	0	0	0	0	21 358	16
	T*(sec)	0.023	0.026	0.039	0.039	0.036	0.046	0.045	2.373	0.070
OPTIMAL SOLUTION	S*	25	25	25	25	25	25	25	25	25
	Nb Nœuds	51 992	2 847	3 156	17 529	17 529	17 529	8 728	21 894	24
	Backtracks	103 921	5 642	6 252	34 996	34 996	34 996	17 404	43 766	31
	Fails	51 960	2 823	3 128	17 496	17 496	17 496	8 697	21 883	16
	T*(sec)	1.936	0.346	0.440	1.352	1.591	1.966	1.523	2.415	0.070
PROOF	Nb Nœuds	4 817 732	327 391	188 107	78 569	78 569	78 569	38 879	26 079	4 994
	Backtracks	9 635 453	654 777	376 409	157 113	157 113	157 113	77 733	52 155	9 987
	Fails	4 817 721	327 386	188 202	78 544	78 544	78 544	38 854	26 076	4 993
	TT(sec)	148.450	9.395	7.154	3.787	4.583	5.740	4.263	2.770	1.361

Start from one initial solution

		V1	V2	V3	V4	V5	V6	V7	V8	V9	V10
SYMETRIE			×	×	×	×	×	×	×	×	×
LINK a AND s				×	×	×	×	×	×	×	×
BRANCH Y					×	×	×	×			
CUMULATIVE						×	×	×	×	×	×
CHANNELING							×	×	×	×	×
CAPASUM								×	×	×	×
DOMOVERWDEG (DPS)									×		
MAXREGRET (DPS)										×	×
SOL EXISTANTE											×
1 SOLUTION	S*	32	27	27	37	37	37	37	26	25	26
	Nb Nœuds	29	28	30	7	7	7	7	21 367	24	10
	Backtracks	3	1	1	0	0	0	0	42 715	31	0
	Fails	3	1	1	0	0	0	0	21 358	16	0
	T*(sec)	0.023	0.026	0.039	0.039	0.036	0.046	0.045	2.373	0.070	0.018
OPTIMAL SOLUTION	S*	25	25	25	25	25	25	25	25	25	25
	Nb Nœuds	51 992	2 847	3 156	17 529	17 529	17 529	8 728	21 894	24	1 213
	Backtracks	103 921	5 642	6 252	34 996	34 996	34 996	17 404	43 766	31	2 405
	Fails	51 960	2 823	3 128	17 496	17 496	17 496	8 697	21 883	16	1 203
	T*(sec)	10936	00346	0.440	1.352	1.591	1.966	1.523	2.415	0.070	0.324
PROOF	Nb Nœuds	4 817 732	327 391	188 107	78 569	78 569	78 569	38 879	26 079	4 994	6 050
	Backtracks	9 635 453	654 777	376 409	157 113	157 113	157 113	77 733	52 155	9 987	12 097
	Fails	4 817 721	327 386	188 202	78 544	78 544	78 544	38 854	26 076	4 993	6 047
	TT(sec)	148.450	9.395	7.154	3.787	4.583	5.740	4.263	2.770	1.361	0.952

Concluding remarks

- CP modelling look like very simple, but can be dangerous due to proximity to ILP modelling with a totally different behaviour of dedicated (powerful) solvers
- CP Good practice
    1. Symmetry breaks
    2. Only branch on decision variables
    3. Promote global constraints
    4. Avoid « bad » constraint with poor propagation
    5. Experiment Redundant constraints
    6. Adapt Search to your problem specific strategies

Concluding remarks

- CP modelling look like very simple, but can be dangerous due to proximity to ILP modelling with a totally different behaviour of dedicated (powerful) solvers
- CP Good practice
    1. Symmetry breaks
    2. Only branch on decision variables
    3. Promote global constraints
    4. Avoid « bad » constraint with poor propagation
    5. Experiment Redundant constraints
    6. Adapt Search to your problem specific strategies

Concluding remarks

- CP modelling look like very simple, but can be dangerous due to proximity to ILP modelling with a totally different behaviour of dedicated (powerful) solvers
- CP Good practice
    1. Symmetry breaks
    2. Only branch on decision variables
    3. Promote global constraints
    4. Avoid « bad » constraint with poor propagation
    5. Experiment Redundant constraints
    6. Adapt Search to your problem specific strategies