LEARNING OPTIMAL DECISION TREES USING CONSTRAINT PROGRAMMING

IJCAI20

Hélène Verhaeghe[1], Siegfried Nijssen[1], Gilles Pesant[2], Claude-Guy Quimper[3], and Pierre Schaus[1]

January 2021

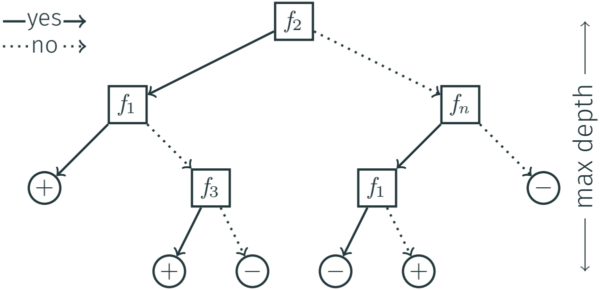
[1]ICTEAM, UCLouvain, Place Sainte Barbe 2, 1348 Louvain-la-Neuve, Belgium, $\{firstname.lastname\}@uclouvain.be$
[2]Polytechnique Montréal, Montréal, Canada, $gilles.pesant@polymtl.ca$
[3]Université Laval, Québec, Canada, $claude-guy.quimper@ift.ulaval.ca$

Database

f_1	f_2	f_3	$\dots$	f_n	c
1	0	1	$\dots$	1	$+$
0	1	0	$\dots$	1	$-$
1	1	0	$\dots$	0	$+$
0	0	0	$\dots$	0	$+$
1	0	0	$\dots$	0	$+$
0	1	1	$\dots$	1	$-$
1	1	1	$\dots$	0	$-$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
1	1	1	$\dots$	1	$+$

Database

f_1	f_2	f_3	$\ldots$	f_n	c
1	0	1	$\ldots$	1	$+$
0	1	0	$\ldots$	1	$-$
1	1	0	$\ldots$	0	$+$
0	0	0	$\ldots$	0	$+$
1	0	0	$\ldots$	0	$+$
0	1	1	$\ldots$	1	$-$
1	1	1	$\ldots$	0	$-$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
1	1	1	$\ldots$	1	$+$



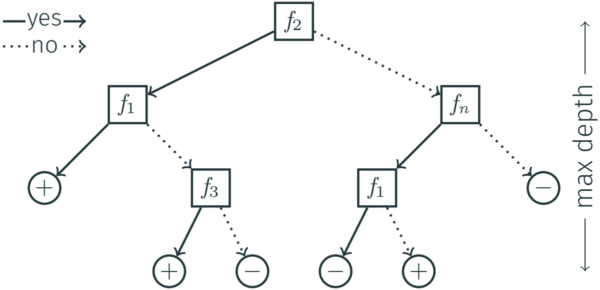
$$\min \sum (pred(i) - c(i))$$

Database

f_1	f_2	f_3	$\dots$	f_n	c
1	0	1	$\dots$	1	$+$
0	1	0	$\dots$	1	$-$
1	1	0	$\dots$	0	$+$
0	0	0	$\dots$	0	$+$
1	0	0	$\dots$	0	$+$
0	1	1	$\dots$	1	$-$
1	1	1	$\dots$	0	$-$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
1	1	1	$\dots$	1	$+$

New sample

0	0	1	$\dots$	0	?



$$\min \sum (pred(i) - c(i))$$

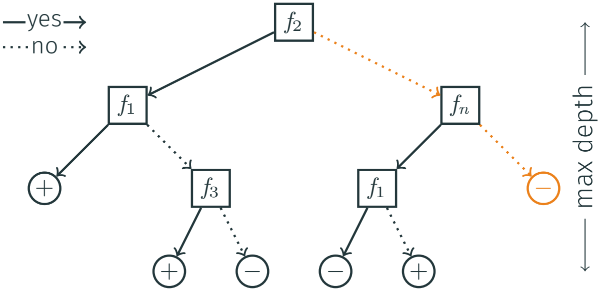
UCLouvain

Database

f_1	f_2	f_3	$\dots$	f_n	c
1	0	1	...	1	+
0	1	0	...	1	−
1	1	0	...	0	+
0	0	0	...	0	+
1	0	0	...	0	+
0	1	1	...	1	−
1	1	1	...	0	−
⋮	⋮	⋮	⋱	⋮	⋮
1	1	1	...	1	+

New sample

0	0	1	...	0	−



$$\min \sum (pred(i) - c(i))$$

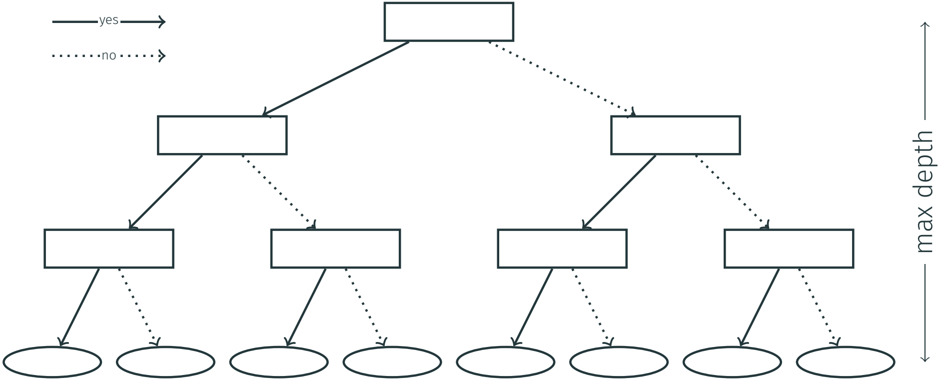
Greedy methods:

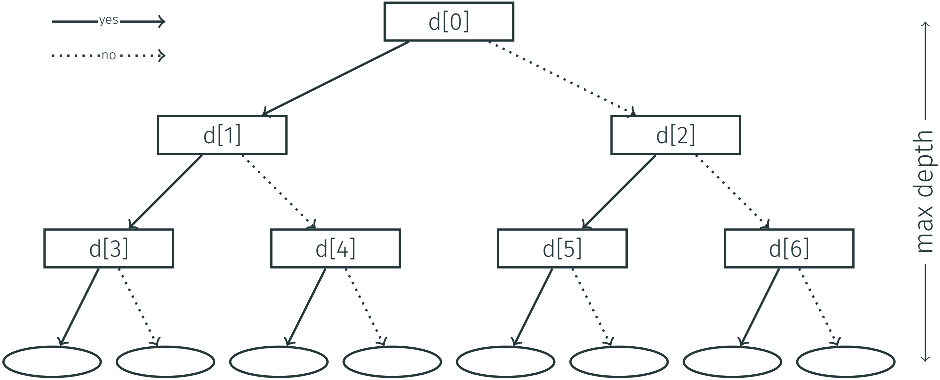
- ✔ easy construction
- ✗ hard to impose additional constraints
- ✗ potentially unnecessarily complex tree

- Mining optimal decision trees from itemset lattices, Nijssen, S., Fromont, E., 2007
- Minimising decision tree size as combinatorial optimisation, Bessiere, C., Hebrard, E., O'Sullivan, B., 2009
- Optimal constraint-based decision tree induction from itemset lattices,Nijssen, S., Fromont, É., 2010
- **Optimal classification trees**, Bertsimas, D., Dunn, J., 2017
- Learning optimal decision trees with sat, Narodytska, N., Ignatiev, A., Pereira, F., Marques-Silva, J., RAS, I., 2018
- Learning optimal and fair decision trees for non-discriminative decision-making, Aghaei, S., Azizi, M.J., Vayanos, P., 2019
- Learning optimal classification trees using a binary linear program formulation, Verwer, S., Zhang, Y., 2019

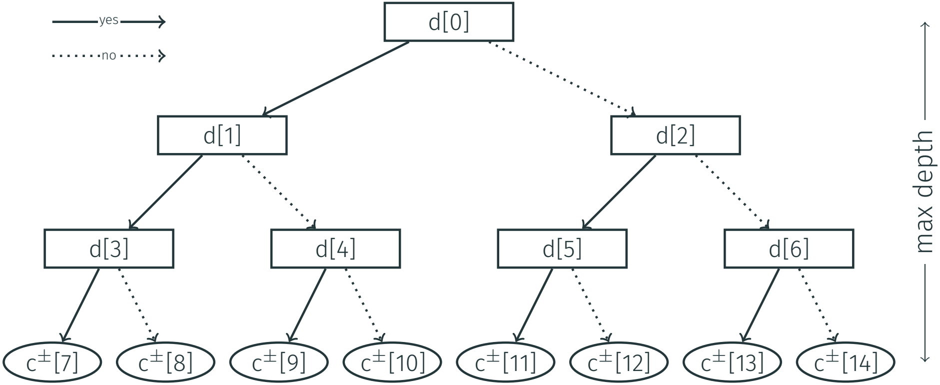
CP MODEL

MODEL - DECISION VARIABLES





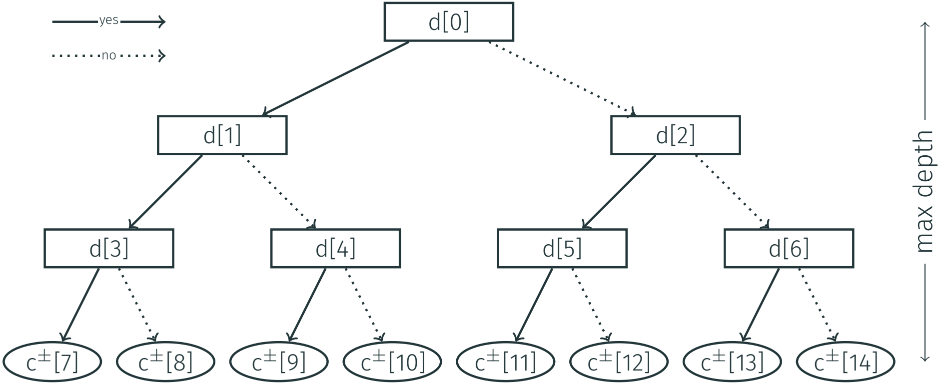
$$dom(d[i]) = \{1, ..., n\}$$



$$dom(d[i]) = \{1, ..., n\}$$

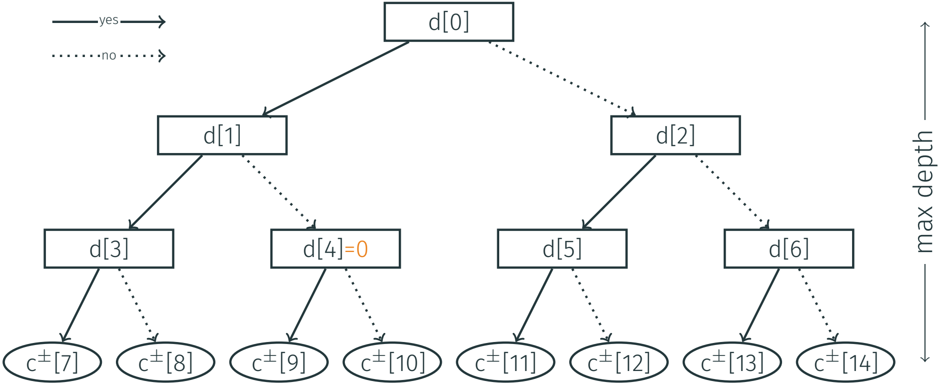
$$dom(c[i]) = \{0, ..., N\}$$

UCLouvain



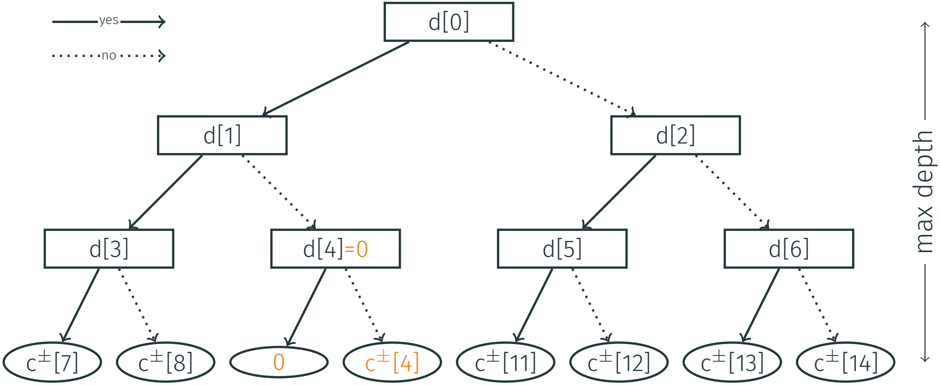
$dom(d[i]) = \{0, 1, ..., n\}$

$dom(c[i]) = \{0, ..., N\}$



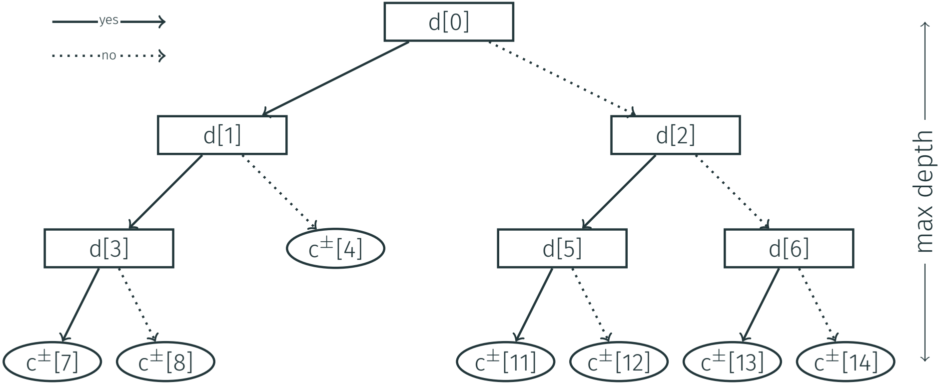
$dom(d[i]) = \{0, 1, ..., n\}$

$dom(c[i]) = \{0, ..., N\}$



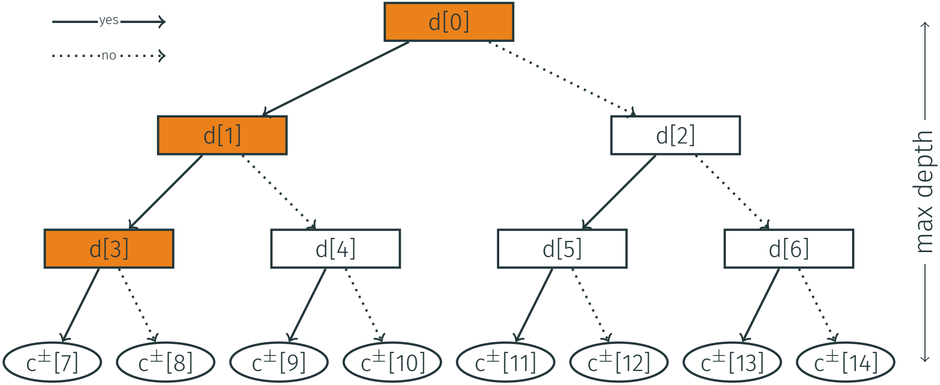
$dom(d[i]) = \{0, 1, ..., n\}$

$dom(c[i]) = \{0, ..., N\}$



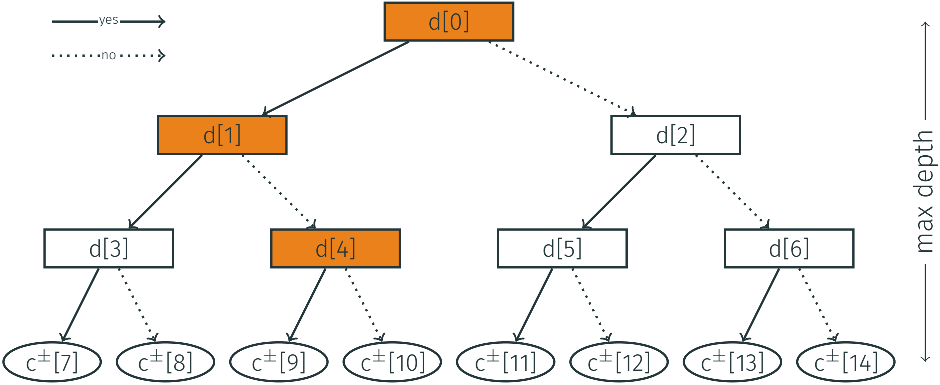
$$dom(d[i]) = \{0, 1, ..., n\}$$

$$dom(c[i]) = \{0, ..., N\}$$



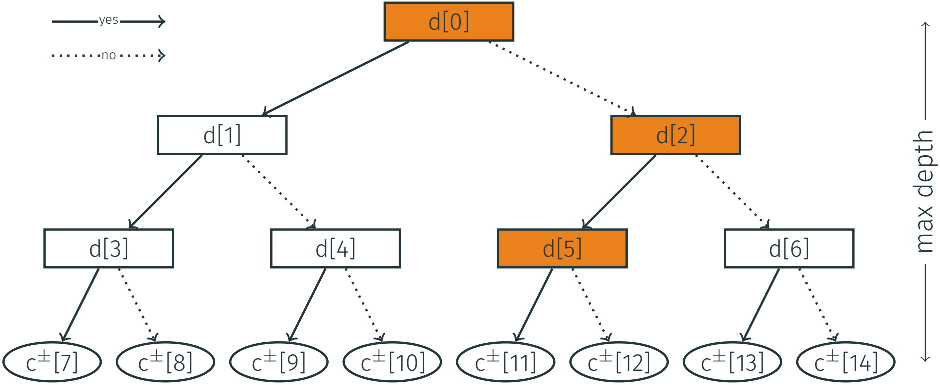
$dom(d[i]) = \{0, 1, ..., n\}$

$dom(c[i]) = \{0, ..., N\}$



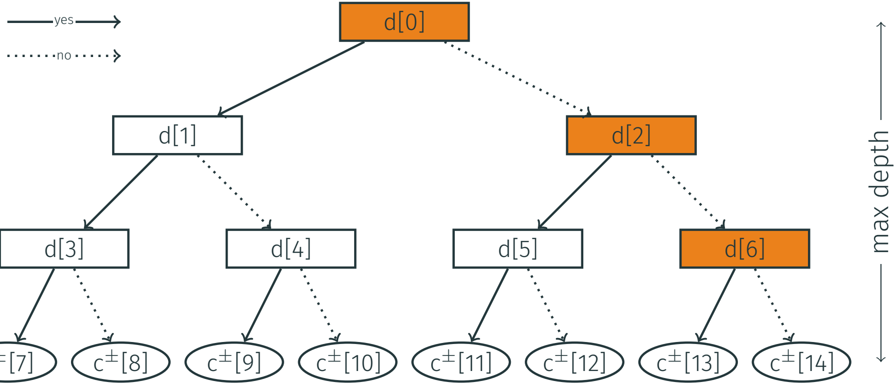
$dom(d[i]) = \{0, 1, ..., n\}$

$dom(c[i]) = \{0, ..., N\}$



$dom(d[i]) = \{0, 1, ..., n\}$

$dom(c[i]) = \{0, ..., N\}$



$dom(d[i]) = \{0, 1, ..., n\}$

$dom(c[i]) = \{0, ..., N\}$

f_1	f_2	f_3	f_4
1	0	1	1
0	1	0	1
1	1	0	0
0	0	0	0
1	0	0	0
0	1	1	1
1	1	1	0
1	1	1	1

Features (Dense)				Counter
x_1	x_2	x_3	x_4	

f_1	f_2	f_3	f_4
1	0	1	1
0	1	0	1
1	1	0	0
0	0	0	0
1	0	0	0
0	1	1	1
1	1	1	0
1	1	1	1

Features (Dense)				Counter
x_1	x_2	x_3	x_4	
0	1	0	1	

f_1	f_2	f_3	f_4
1	0	1	1
0	1	0	1
1	1	0	0
0	0	0	0
1	0	0	0
0	1	1	1
1	1	1	0
1	1	1	1

Features (Dense)				Counter
x_1	x_2	x_3	x_4	
0	1	0	1	3

f_1	f_2	f_3	f_4
1	0	1	1
0	1	0	1
1	1	0	0
0	0	0	0
1	0	0	0
0	1	1	1
1	1	1	0
1	1	1	1

Features (Dense)				Counter
x_1	x_2	x_3	x_4	
0	1	0	1	3

- Dense representation
- No feature rejection

f_1	f_2	f_3	f_4
1	0	1	1
0	1	0	1
1	1	0	0
0	0	0	0
1	0	0	0
0	1	1	1
1	1	1	0
1	1	1	1

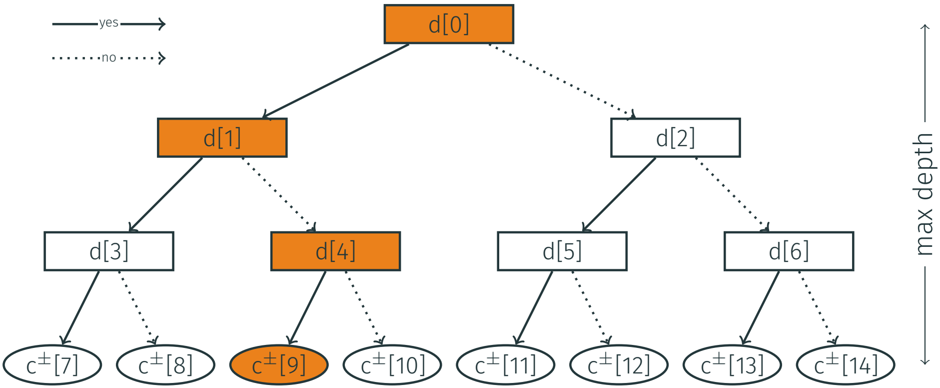
Features (Sparse)		Counter
y_1	y_2	
2	4	3

- ~~Dense representation~~
- No feature rejection

f_1	f_2	f_3	f_4
1	0	1	1
0	1	0	1
1	1	0	0
0	0	0	0
1	0	0	0
0	1	1	1
1	1	1	0
1	1	1	1

✔Features (Sparse)		✗Features (Sparse)	Counter
y_1	y_2	z_1	
2	4	3	1

- ~~Dense representation~~
- ~~No feature rejection~~



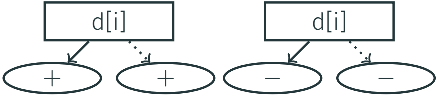
$Coversize(\{d[0], d[4]\}, \{d[1]\}, c^+[9])$

$Coversize(\{d[0], d[4]\}, \{d[1]\}, c^-[9])$

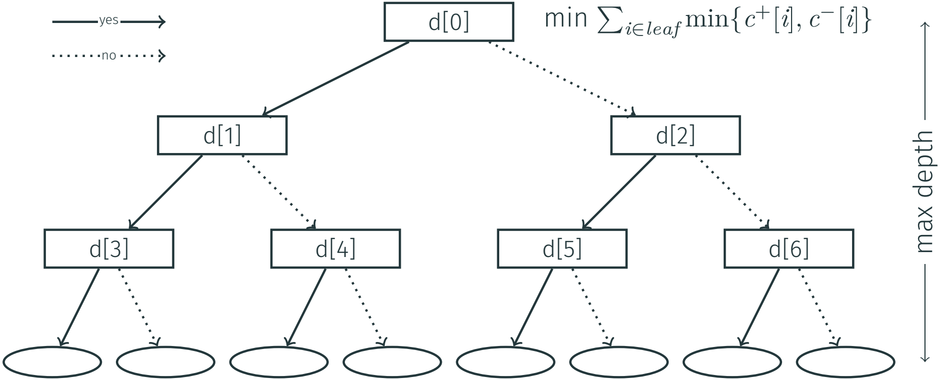
- constraints imposing minimum at leaf

$$c^+[i] + c^-[i] \geq N_{min}$$

- constraints avoiding useless decisions



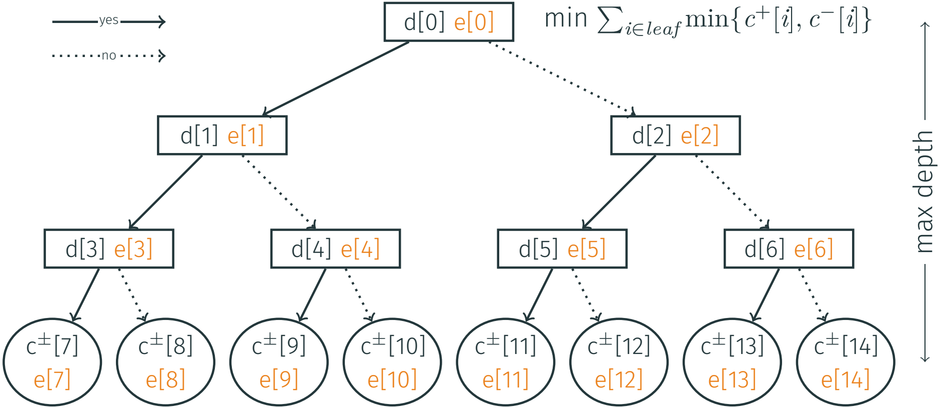
- redundant constraints improving speed



min $\sum_{i \in leaf} \min\{c^+[i], c^-[i]\}$

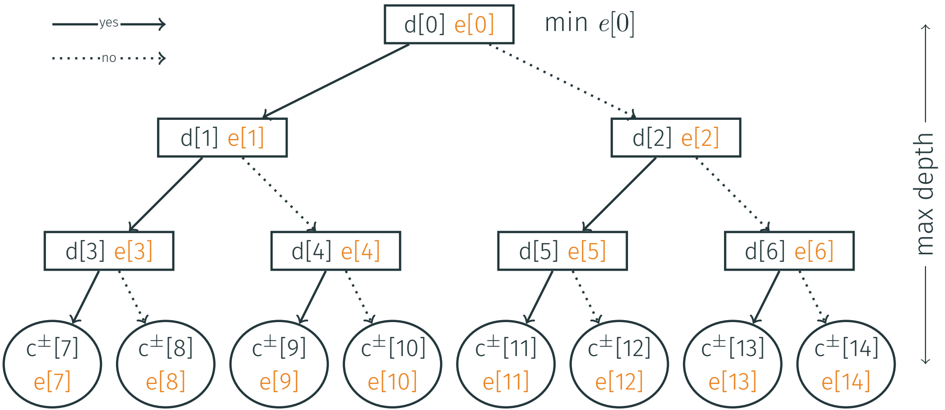
$dom(d[i]) = \{0, 1, ..., n\}$

$dom(c[i]) = \{0, ..., N\}$



min $\sum_{i \in leaf} \min\{c^+[i], c^-[i]\}$

$dom(d[i]) = \{0, 1, ..., n\}$ $dom(c[i]) = \{0, ..., N\}$ $dom(e[i]) = \{0, ..., N\}$



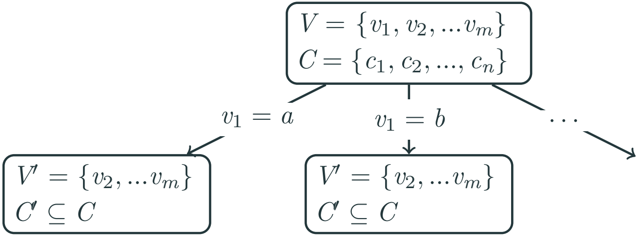
$dom(d[i]) = \{0, 1, ..., n\}$ $dom(c[i]) = \{0, ..., N\}$ $dom(e[i]) = \{0, ..., N\}$

SEARCH

$$V = \{v_1, v_2, ... v_m\}$$
$$C = \{c_1, c_2, ..., c_n\}$$

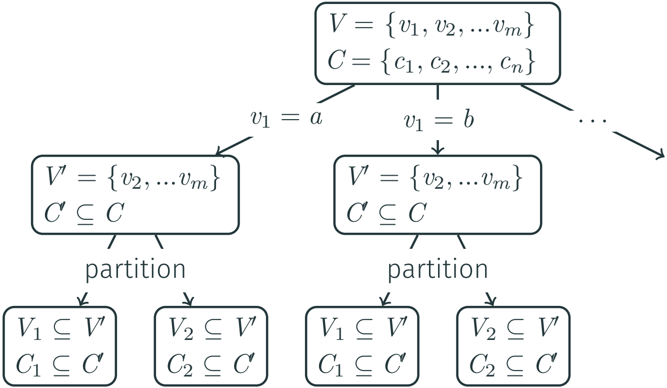
OR nodes

SOL = SOL_1 or SOL_2 or ...



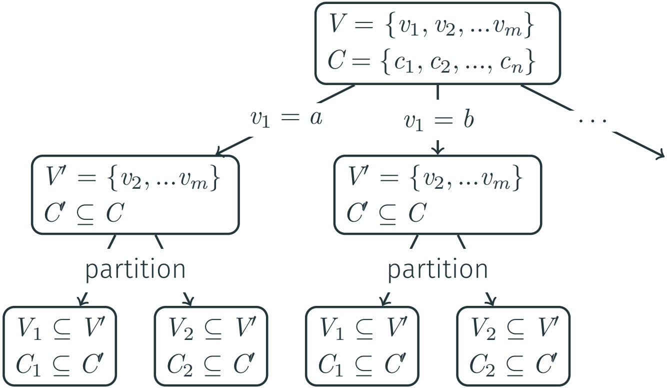
OR nodes

SOL = SOL_1 or SOL_2 or ...



OR nodes

SOL = SOL_1 or SOL_2 or ...

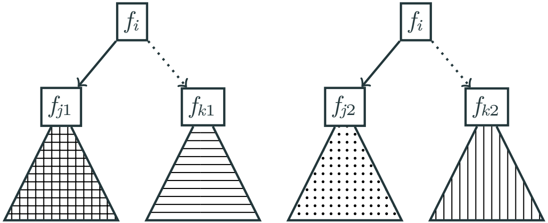


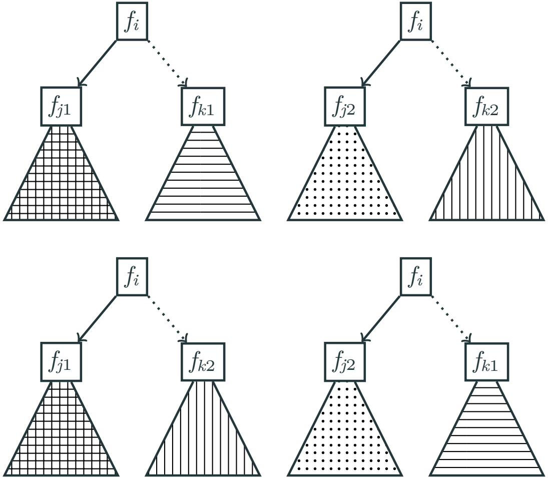
OR nodes

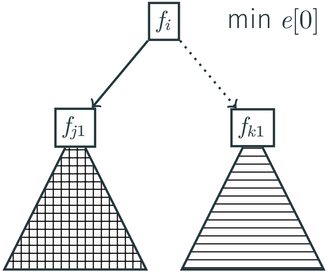
SOL = SOL_1 or SOL_2 or $\dots$

AND nodes

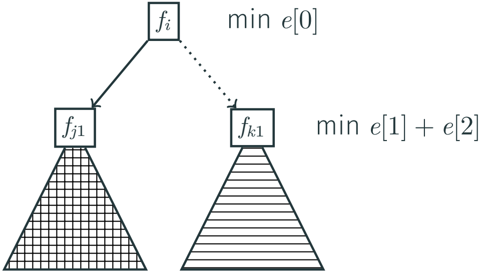
SOL = SOL_1 and SOL_2 and $\dots$

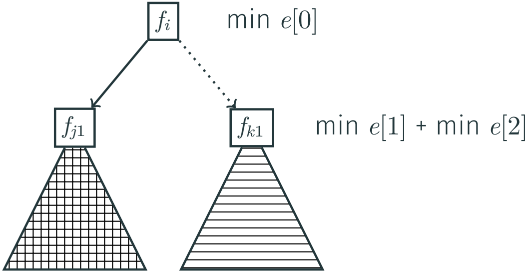

f_i
f_{j1}
f_{k1}
f_i
f_{j2}
f_{k2}


f_i
f_{j1}
f_{k1}
f_i
f_{j2}
f_{k2}
f_i
f_{j1}
f_{k2}
f_i
f_{j2}
f_{k1}

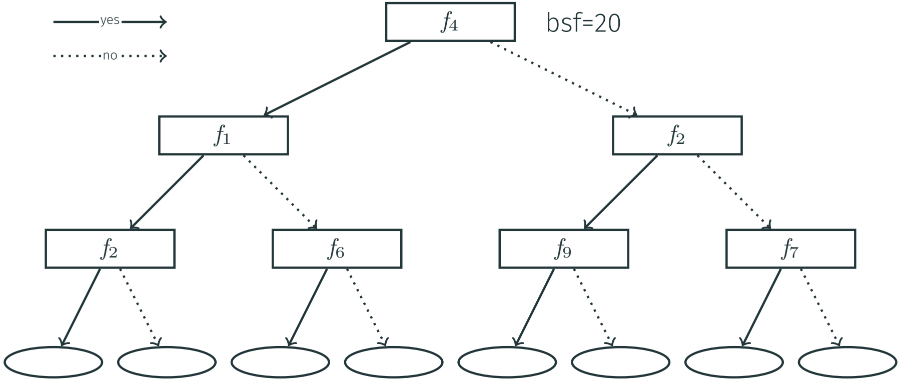

f_i
min $e[0]$
f_{j1}
f_{k1}

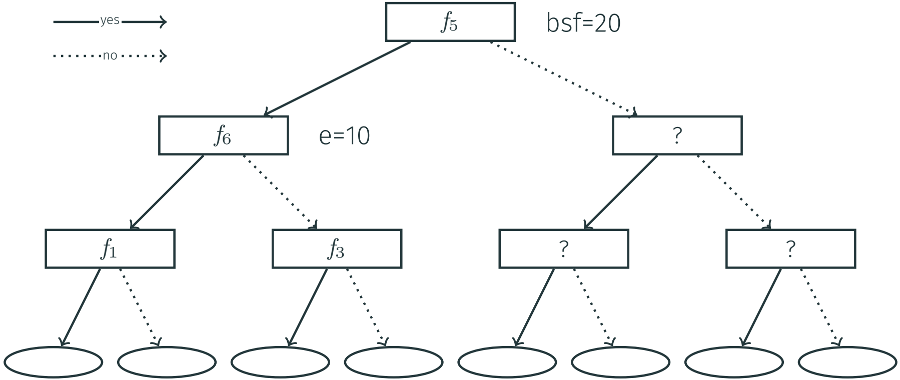
AND/OR SEARCH TREE

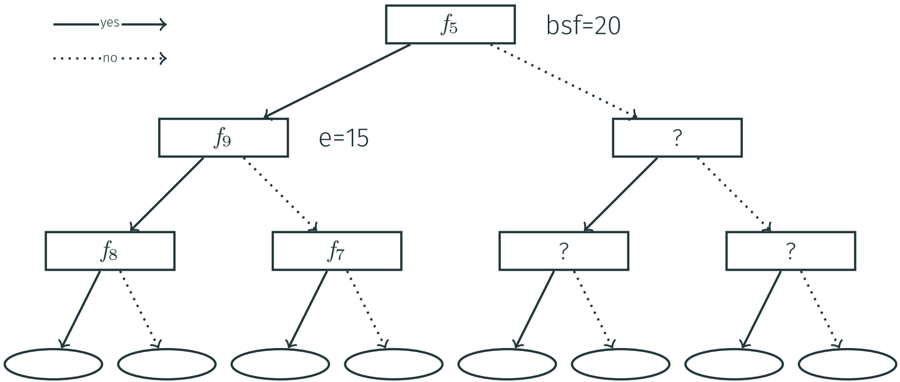


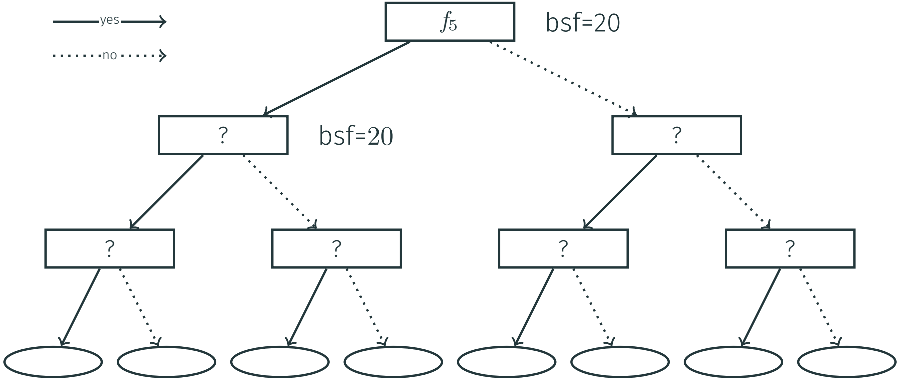


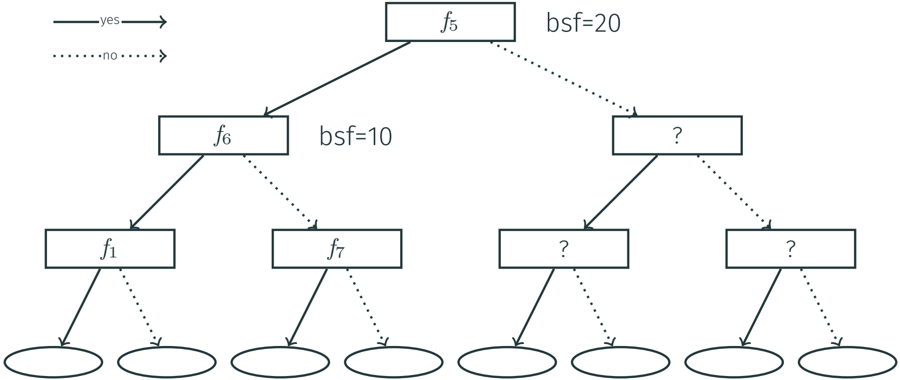
f_i
min $e[0]$
f_{j1}
f_{k1}
min $e[1]$ + min $e[2]$

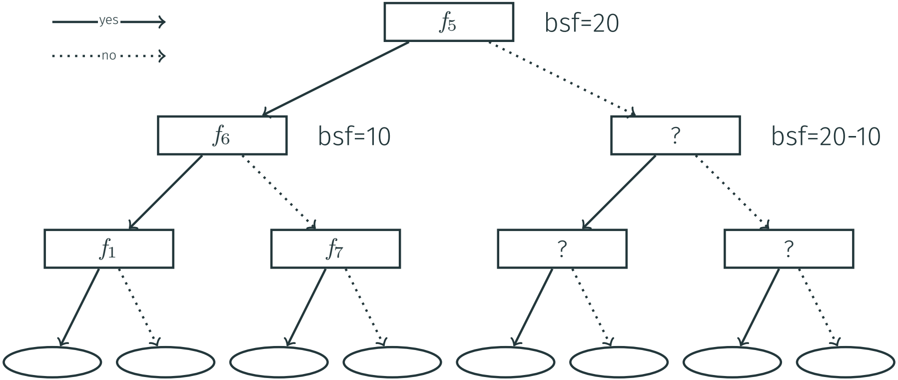

yes
no
f_4
bsf=20
f_1
f_2
f_2
f_6
f_9
f_7

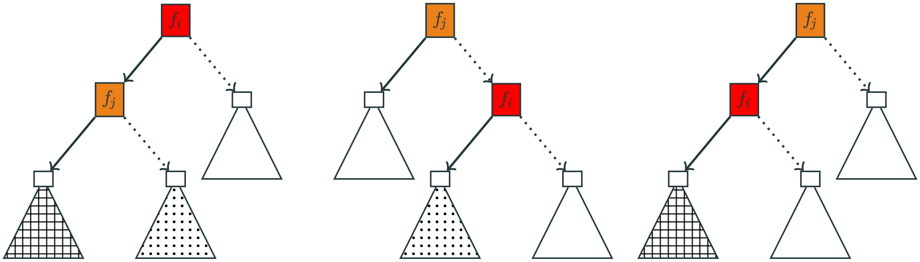

yes
no
f_5
bsf=20
f_6
e=10
?
f_1
f_3
?
?

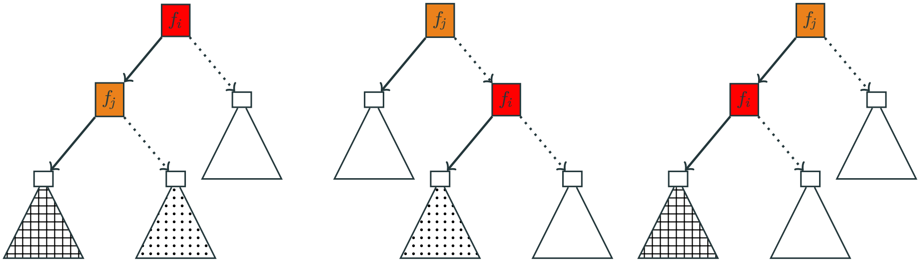

yes
no
f_5
bsf=20
f_9
e=15
?
f_8
f_7
?
?

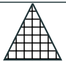
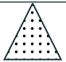

yes
no
f_5
bsf=20
?
bsf=20
?
?
?
?
?


yes
no
f_5
bsf=20
f_6
bsf=10
?
f_1
f_7
?
?


yes
no
f_5
bsf=20
f_6
bsf=10
?
bsf=20-10
f_1
f_7
?
?


f_i
f_j
f_j
f_i
f_j
f_i



	yes	no	hash
	f_i f_j		$f_i, f_j -$
	f_i	f_j	$f_i - f_j$

RESULTS

	$N_{\min} = 1$			$N_{\min} = 5$			
	DL8	BinOCT	CP	DL8	CP	CP-c	CP-m
Proven optimality	49(64%)	13(17%)	**57**(75%)	54(71%)	56(74%)	56(74%)	**58**(76%)
Best solution found	49(64%)	21(28%)	**76**(100%)	54(71%)	**74**(97%)	**74**(97%)	70(92%)
Fastest	23(30%)	11(14%)	**49**(64%)	28(37%)	**40**(53%)	33(43%)	22(29%)
Time out	27(36%)	63(83%)	**19**(25%)	22(29%)	21(28%)	21(28%)	**19**(25%)

23 instances, depths from 2 to 5, 10 min TO

DL8: Dynamic programming approach using frequent itemsets mining

BinOCT: MIP-based approach running on CPLEX

To summarize

- efficient method
- cp based
- exploits the structure of the problem
- anytime best solution

To go further

- multi-class decision trees
- continuous features through binarization
- other sum-based cost functions
- ...

Thank you for listening!

Any questions?

Also, our extended journal paper is out!