Soft constraint processing

Javier Larrosa
UPC - Barcelona
Spain

Thomas Schiex
INRA - Toulouse
France

Overview

1. **Frameworks**
   - **Generic and specific**
2. **Algorithms**
   - **Search: complete and incomplete**
   - **Inference: complete and incomplete**
3. **Integration with CP**
   - **Soft as hard**
   - **Soft as global constraint**

Parallel mini-tutorial

- CSP $\Leftrightarrow$ SAT strong relation
- Along the presentation, we will highlight the connections with SAT

- Multimedia trick:
  - SAT slides in yellow background

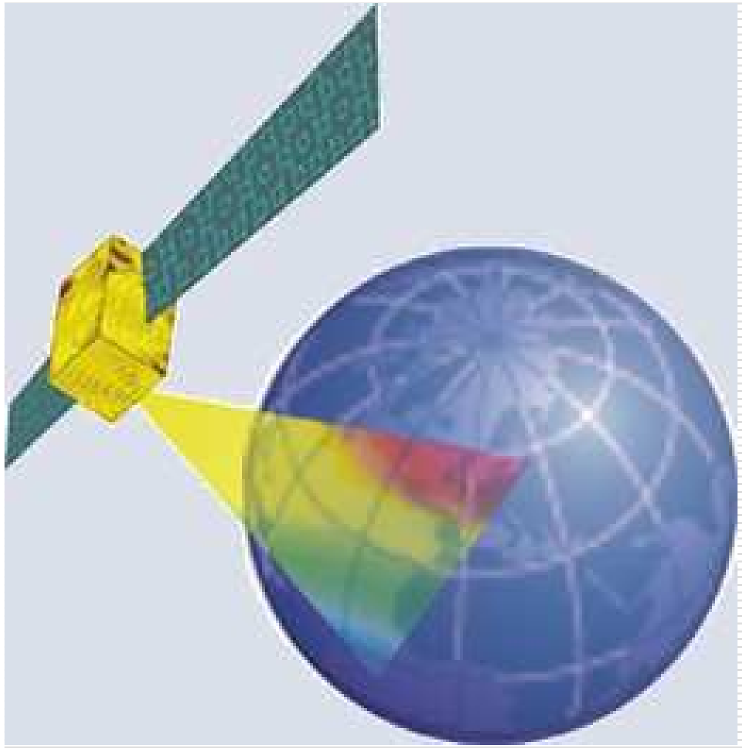
Why soft constraints?

- CSP framework: natural for *decision* problems
- SAT framework: natural for *decision* problems with *boolean* variables

- Many problems are *constrained optimization* problems and the difficulty is in the optimization part

Why soft constraints?

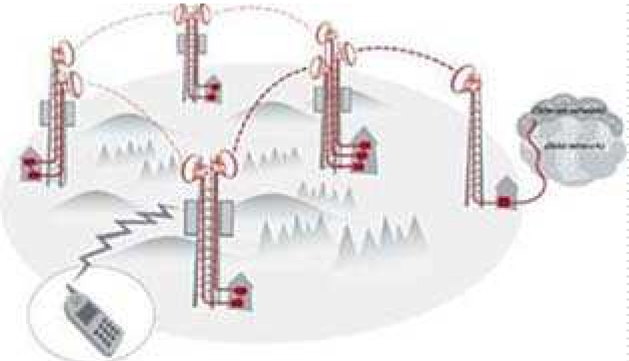
- Earth Observation Satellite Scheduling



  - Given a set of requested pictures (of different importance)…
  - … select the best subset of compatible pictures …
  - … subject to available resources:
    - 3 on-board cameras
    - Data-bus bandwith, setup-times, orbiting
  - Best = maximize sum of importance

Why soft constraints?

- Frequency assignment

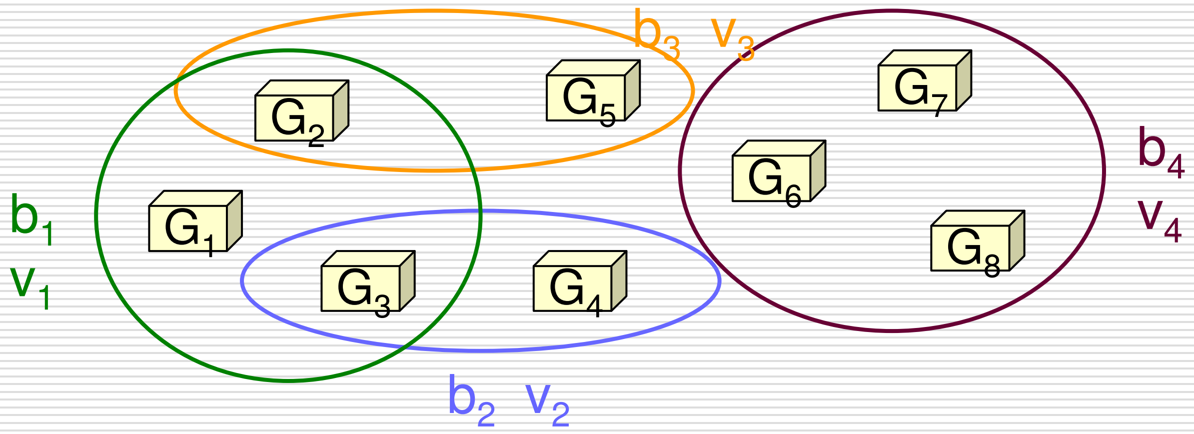


- Given a telecommunication network
- ...find the best frequency for each communication link avoiding interferences

- Best can be:
  - Minimize the maximum frequency (max)
  - Minimize the global interference (sum)

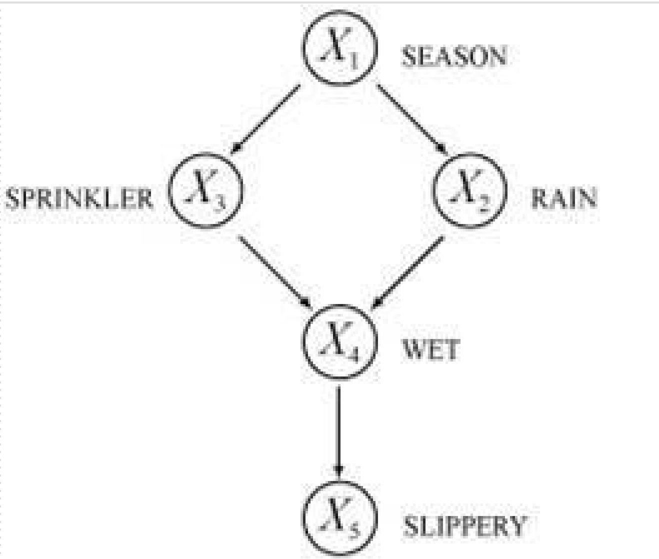
Why soft constraints?

- Combinatorial auctions
  - Given a set G of goods and a set B of bids…
    - Bid (b_i, v_i), b_i requested goods, v_i value
  - … find the best subset of compatible bids
  - Best = maximize revenue (sum)



Why soft constraints?

- Probabilistic inference (bayesian nets)



  - Given a probability distribution defined by a DAG of conditional probability tables
  - and some evidence …
  - …find the *most probable* explanation for the evidence (product)

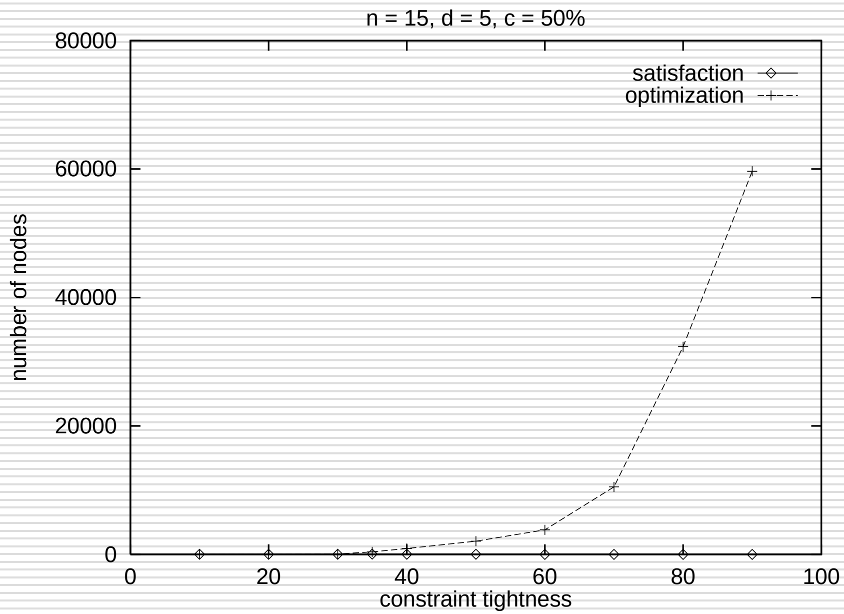
Why soft constraints?

- Even in decision problems:
  - users may have *preferences* among solutions

Experiment: give users a few solutions and they will find reasons to prefer some of them.

Observation

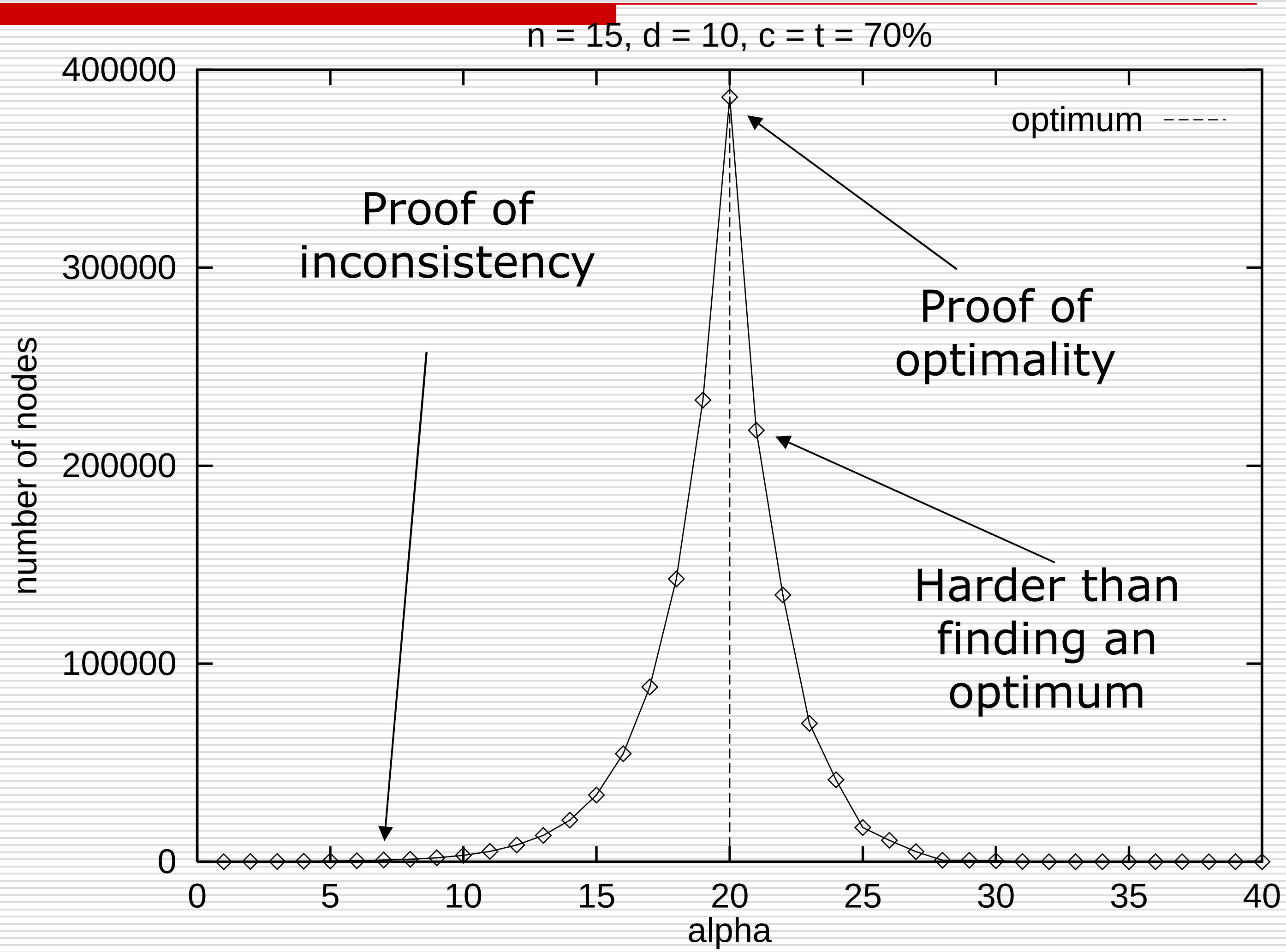
- Optimization problems are harder than satisfaction problems



CSP vs. Max-CSP

Why is it so hard ?

Problem P(alpha): is there an assignment of cost lower than alpha ?



Notation

- $X=\{x_1,\ldots, x_n\}$ variables (n variables)
- $D=\{D_1,\ldots, D_n\}$ finite domains (max size d)

- $Z \subseteq Y \subseteq X$,
  - t_Y is a tuple on Y
  - $t_Y[Z]$ is its projection on Z
  - $t_Y[-x] = t_Y[Y-\{x\}]$ is projecting out variable x
  - $f_Y: \prod_{x_i \in Y} D_i \rightarrow E$ is a cost function on Y

Generic and specific frameworks

Valued CN	weighted CN
Semiring CN	fuzzy CN
	...

Costs (preferences)

- E costs (preferences) set
  - ordered by $\preccurlyeq$
  - if $a \preccurlyeq b$ then a is better than b
- Costs are associated to tuples
- Combined with a dedicated *operator* $\oplus$
  - *max*: priorities — Fuzzy/possibilistic CN
  - +: additive costs — Weighted CN
  - *: factorized probabilities… — Probabilistic CN, BN

Soft constraint network (CN)

- (X,D,C)
  - $X=\{x_1,\ldots, x_n\}$ variables
  - $D=\{D_1,\ldots, D_n\}$ finite domains
  - $C=\{f,\ldots\}$ cost functions
    - f_S, f_{ij}, f_i $f_\varnothing$ scope $S,\{x_i,x_j\},\{x_i\}, \varnothing$
    - $f_S(t)$: $\rightarrow$ E (ordered by $\preccurlyeq$, $\perp \preccurlyeq$ T)

identity

anihilator

- commutative
- associative
- monotonic

- **Obj. Function**: $F(X)= \oplus f_S\ (X[S])$
- **Solution**: $F(t) \neq$ T
- **Task**: find optimal solution

Specific frameworks

Instance	E	$\oplus$	$\bot \preccurlyeq \top$
Classic CN	$\{t,f\}$	and	$t \preccurlyeq f$
Possibilistic	[0,1]	max	$0 \preccurlyeq 1$
Fuzzy CN	[0,1]	$\max_{\preccurlyeq}$	$1 \preccurlyeq 0$
Weighted CN	[0,k]	$+$	$0 \preccurlyeq k$
Bayes net	[0,1]	$\times$	$1 \preccurlyeq 0$

Weighted Clauses

- (C,w) weighted clause
  - C disjunction of literals
  - w cost of violation
  - $w \in E$ (ordered by $\preccurlyeq$, $\perp \preccurlyeq T$)
  - $\oplus$ combinator of costs
- Cost functions = weighted clauses

x_i	x_j	$f(x_i,x_j)$	
0	0	6	$\longrightarrow$ $(\mathbf{x_i} \vee \mathbf{x_j}, 6)$,
0	1	0	
1	0	2	$\longrightarrow$ $(\neg \mathbf{x_i} \vee \mathbf{x_j}, 2)$,
1	1	3	$\longrightarrow$ $(\neg \mathbf{x_i} \vee \neg \mathbf{x_j}, 3)$

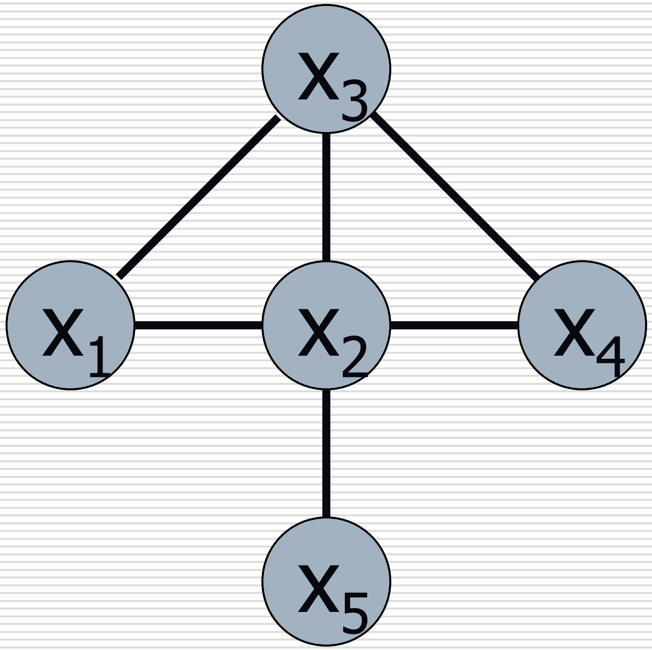
Soft CNF formula

- F={(C,w),…} Set of weighted clauses
  - (C, T) mandatory clause
  - (C, w<T) non-mandatory clause
- **Valuation**: $F(X) = \oplus\ w$ (aggr. of unsatisfied)
- **Model**: $F(t) \neq T$
- **Task:** find optimal model

Specific weighted prop. logics

Instance	E	$\oplus$	$\bot \preccurlyeq \top$
SAT	$\{t,f\}$	and	$t \preccurlyeq f$
Fuzzy SAT	[0,1]	$\max_{\preccurlyeq}$	$1 \preccurlyeq 0$
Max-SAT	[0,k]	$+$	$0 \preccurlyeq k$
Markov Prop. Logic	[0,1]	$\times$	$1 \preccurlyeq 0$

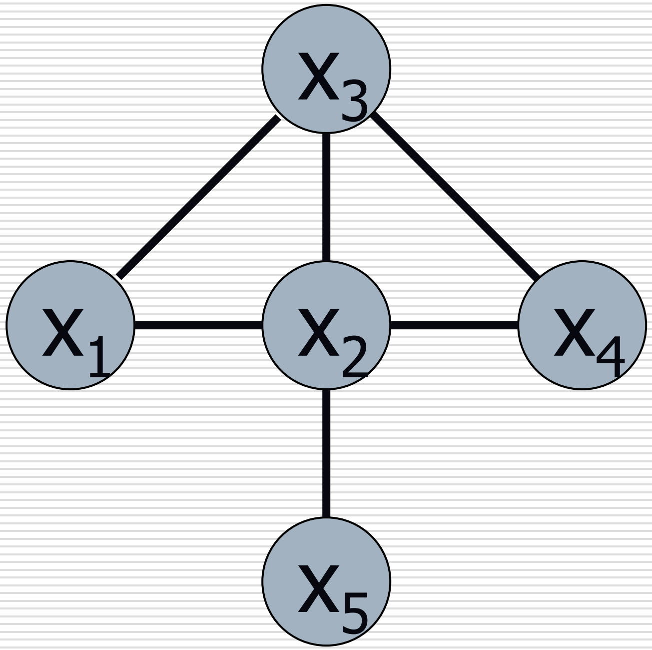
CSP example (3-coloring)



For each edge:
(hard constr.)

x_i	x_j	$f(x_i,x_j)$
b	b	**T**
b	g	$\perp$
b	r	$\perp$
g	b	$\perp$
g	g	**T**
g	r	$\perp$
r	b	$\perp$
r	g	$\perp$
r	r	**T**

Weighted CSP example (⊕ = +)

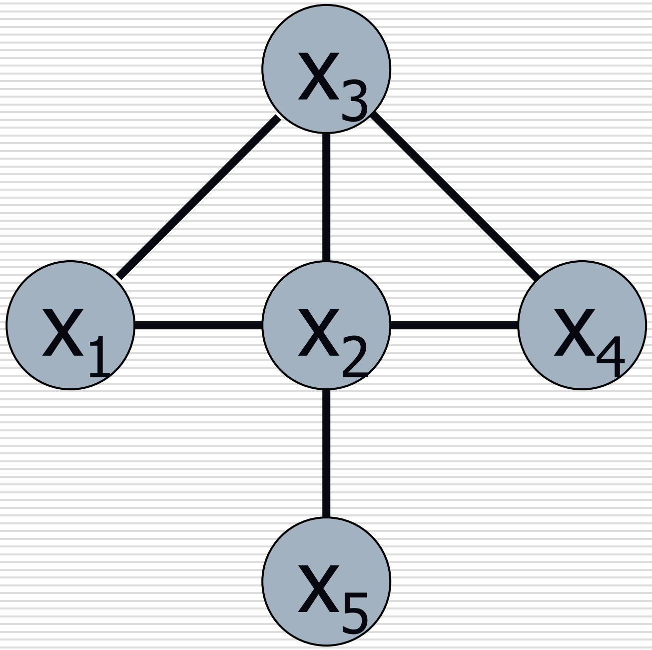


For each vertex

x_i	$f(x_i)$
b	0
g	1
r	1

F(*X*): number of non blue vertices

Possibilistic CSP example (⊕=max)



For each vertex

x_i	$f(x_i)$
b	0.0
g	0.1
r	0.2

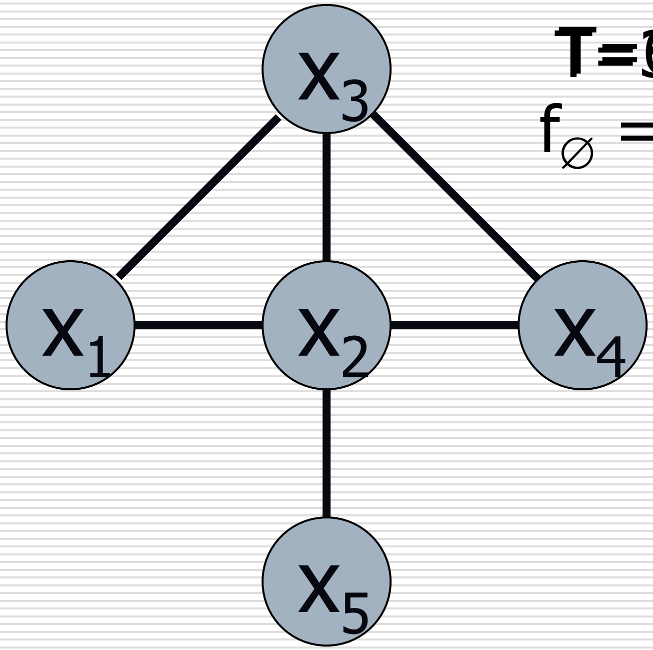
F(*X*): highest color used (b<g<r)

Some important details

- T = maximum acceptable violation.
- Empty scope soft constraint $f_\varnothing$ (a constant)
  - Gives an obvious lower bound on the optimum
  - If you do not like it: $f_\varnothing = \perp$

Additional expression power

Weighted CSP example ($\oplus$ = +)



T=3

$f_\varnothing = 0$

For each vertex

$\mathbf{x_i}$	$\mathbf{f(x_i)}$
b	0
g	1
r	1

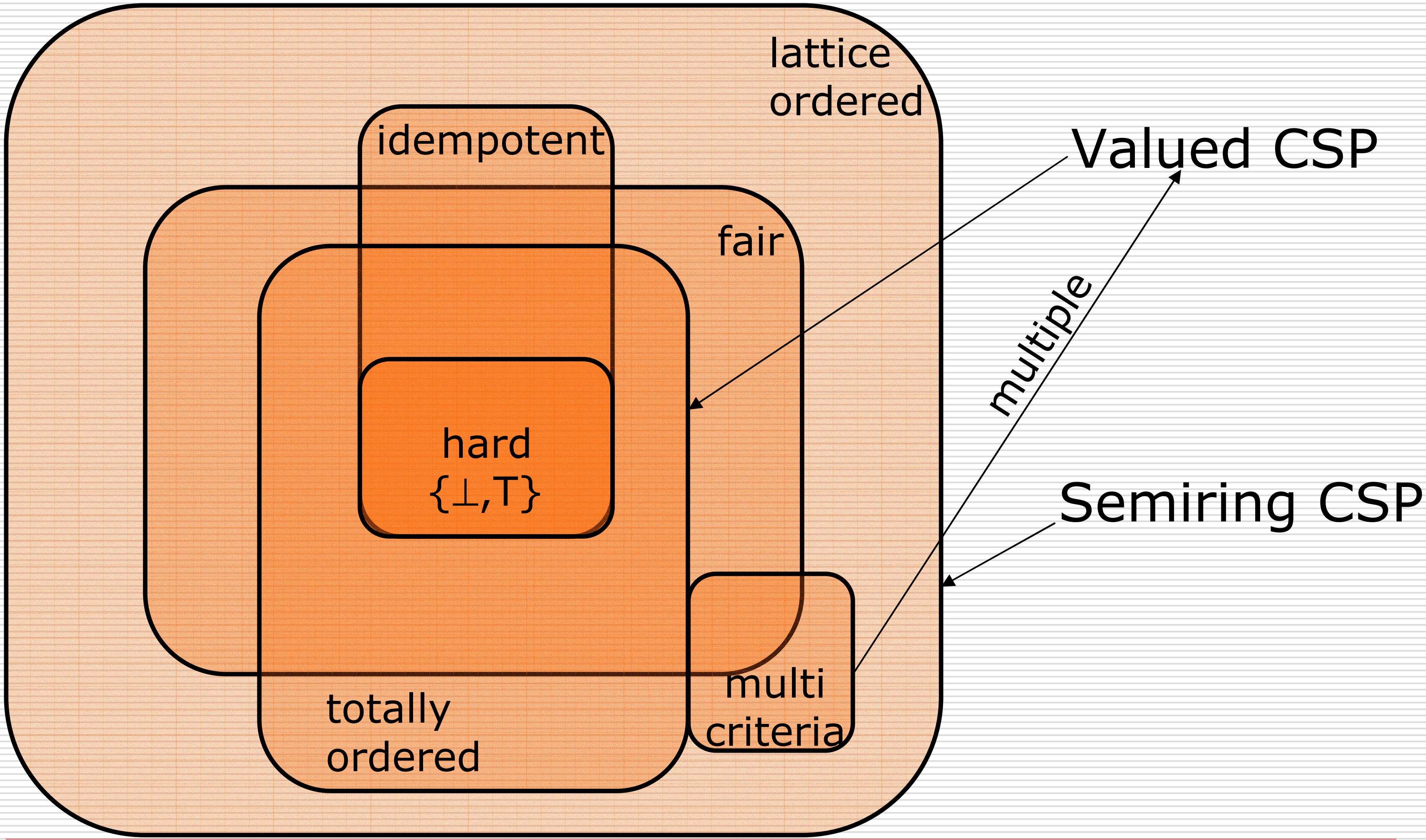
For each edge:

$\mathbf{x_i}$	$\mathbf{x_j}$	$\mathbf{f(x_i,x_j)}$
b	b	**T**
b	g	0
b	r	0
g	b	0
g	g	**T**
g	r	0
r	b	0
r	g	0
r	r	**T**

F(*X*): number of non blue vertices

Optimal coloration with less than 3 non-blue

General frameworks and cost structures



Idempotency

$$a \oplus a = a \text{ (for any } a)$$

$$\Updownarrow$$

$$\text{For any } f_S \text{ implied by } (X,D,C)$$

$$(X,D,C) \equiv (X,D,C \cup \{f_S\})$$

- Classic CN: $\oplus$ = and
- Possibilistic CN: $\oplus$ = max
- Fuzzy CN: $\oplus = \max_{\preccurlyeq}$
- …

Fairness

- Ability to compensate for cost increases by subtraction using a pseudo-difference:

$$\text{For } b \preccurlyeq a,\ (a \ominus b) \oplus b = a$$

  - Classic CN: $a \ominus b$ = or (max)
  - Fuzzy CN: $a \ominus b = \max_{\preccurlyeq}$
  - Weighted CN: $a \ominus b = a\text{-}b$ $(a \neq T)$ else T
  - Bayes nets: $a \ominus b = /$
  - …

Processing Soft constraints

- **Search**
  - **complete (systematic)**
  - **incomplete (local)**
- **Inference**
  - **complete (variable elimination)**
  - **incomplete (local consistency)**

Systematic search

Branch and bound(s)

I - Assignment (conditioning)

x_i	x_j	$f(x_i,x_j)$
b	b	T
b	g	0
b	r	3
g	b	0
g	g	T
g	r	0
r	b	0
r	g	0
r	r	T

$f[x_i=b]$

$g(x_j)$

0
3

$g[x_j=r]$

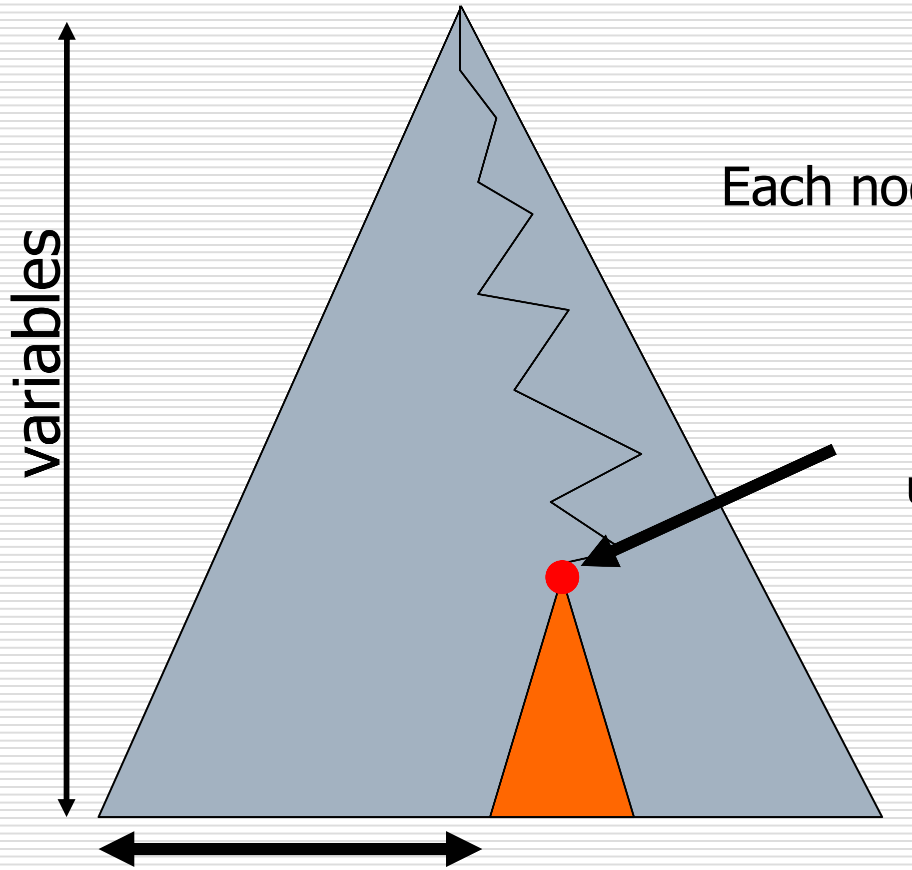
$h_{\varnothing}$

I - Assignment (conditioning)

$$\{(x \vee y \vee z,3),\ (\neg x \vee y,2)\} \xrightarrow{x=\mathit{true}} (y,2) \xrightarrow{y=\mathit{false}} (\quad,2)$$

empty clause.
It cannot be satisfied,
2 is necessary cost

Systematic search



Each node is a soft constraint subproblem

(LB) Lower Bound = $f_\varnothing$
under estimation of the best
solution in the sub-tree

If LB $\geq$ UB then prune

(UB) Upper Bound = best solution so far = **T**

Depth First Search (DFS)

BT(X,D,C)

    if (X=∅) then Top :=$f_\varnothing$

    else

        x_j := selectVar(X)

        forall $a \in D_j$ do

            $\forall_{f_S \in C \text{ s.t. } x_j \in S}\ f := f[x_j = a]$

            $f_\varnothing := \Sigma_{g_S \in C \text{ s.t. } S=\varnothing}\ g_S$

            if ($f_\varnothing$ <Top) then BT(X-{x_j},D-{D_j},C)

variable heuristics

value heuristics

improve LB

good UB ASAP

Improving the lower bound (WCSP)

- Sum up costs that will necessarily occur (no matter what values are assigned to the variables)

- PFC-DAC (Wallace et al. 1994)
- PFC-MRDAC (Larrosa et al. 1999…)
- Russian Doll Search (Verfaillie et al. 1996)
- Mini-buckets (Dechter et al. 1998)

Improving the lower bound (Max-SAT)

- Detect independent subsets of mutually inconsistent clauses

- LB4a (Shen and Zhang, 2004)
- UP (Li et al, 2005)
- Max Solver (Xing and Zhang, 2005)
- MaxSatz (Li et al, 2006)
- …

Local search

Nothing really specific

Local search

Based on perturbation of solutions in a local neighborhood

- Simulated annealing
- Tabu search
- Variable neighborhood search
- Greedy rand. adapt. search (GRASP)
- Evolutionary computation (GA)
- Ant colony optimization…

For boolean variables:
- GSAT
- …

- See: Blum & Roli, ACM comp. surveys, 35(3), 2003

Boosting Systematic Search with Local Search

- Do local search prior systematic search
- Use best cost found as initial T
  - If optimal, we just prove optimality
  - In all cases, we may improve pruning

Boosting Systematic Search with Local Search

- Ex: Frequency assignment problem
  - Instance: CELAR6-sub4
    - #var: 22 , #val: 44 , Optimum: 3230
  - Solver: toolbar 2.2 with default options
  - T initialized to 100000 → 3 hours
  - T initialized to 3230 → 1 hour
    - Optimized local search can find the optimum in a less than 30” (incop)

Complete inference

Variable (bucket) elimination

Graph structural parameters

II - Combination (join with $\oplus$, + here)

x_i	x_j	$f(x_i,x_j)$
b	b	6
b	g	0
g	b	0
g	g	6

$\oplus$

x_j	x_k	$g(x_j,x_k)$
b	b	6
b	g	0
g	b	0
g	g	6

x_i	x_j	x_k	$h(x_i,x_j,x_k)$
b	b	b	12
b	b	g	6
b	g	b	0
b	g	g	6
g	b	b	6
g	b	g	0
g	g	b	6
g	g	g	12

$= 0 \oplus 6$

III - Projection (elimination)

x_i	x_j	$f(x_i,x_j)$
b	b	4
b	g	6
b	r	0
g	b	2
g	g	6
g	r	3
r	b	1
r	g	0
r	r	6

Min

$f[x_i]$

x_i	$g(x_i)$
b	0
g	2
r	0

$g[\varnothing]$

$h_{\varnothing}$
0

Properties

- Replacing two functions by their combination preserves the problem
- If f is the only function involving variable x, replacing f by $f[-x]$ preserves the optimum

Variable elimination

1. Select a variable
2. Sum all functions that mention it
3. Project the variable out

- Complexity

Time: $\Theta(\exp(deg+1))$

Space: $\Theta(\exp(deg))$

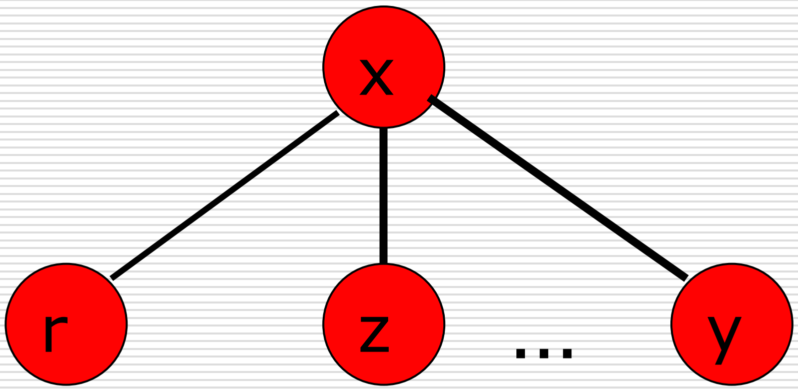
Variable elimination **(aka bucket elimination)**

- Eliminate Variables one by one.
- When all variables have been eliminated, the **problem is solved**
- Optimal solutions of the original problem can be recomputed

•Complexity: exponential in the *induced width*

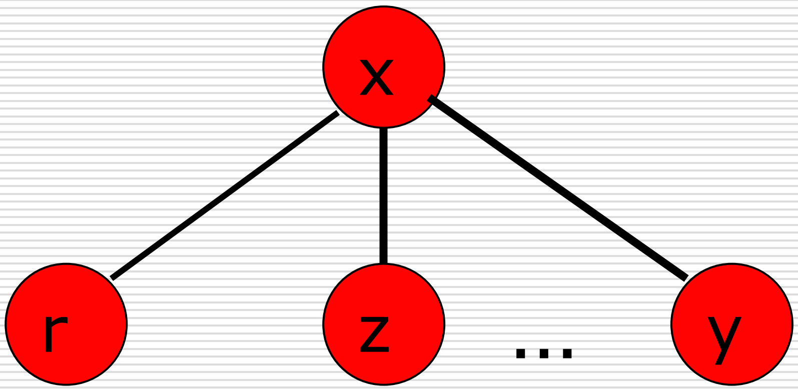
Elimination order influence

- {f(x,r), f(x,z), …, f(x,y)}
- Order: r, z, …, y, x



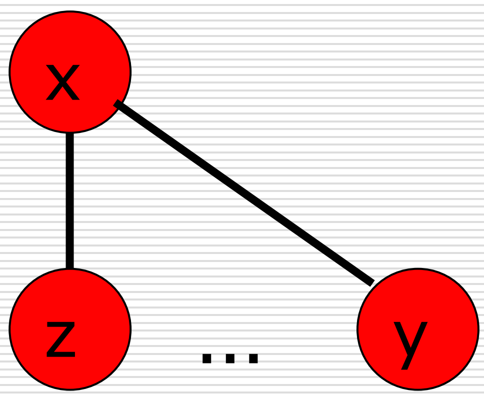
Elimination order influence

- {f(x,r), f(x,z), …, f(x,y)}
- Order: r, z, …, y, x



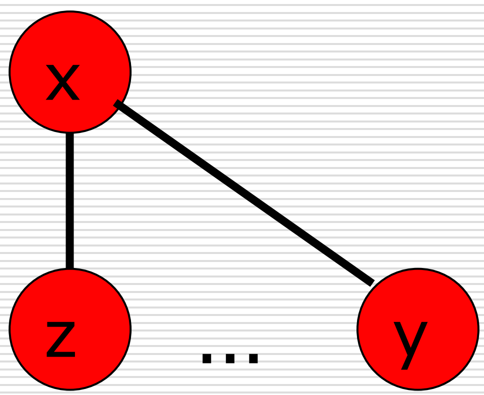
Elimination order influence

- {f(x), f(x,z), …, f(x,y)}
- Order: z, …, y, x



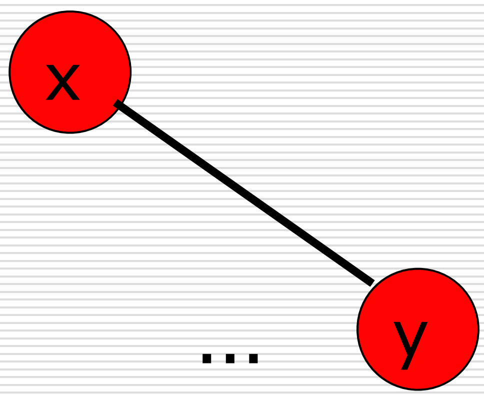
Elimination order influence

- {f(x), f(x,z), …, f(x,y)}
- Order: z, …, y, x



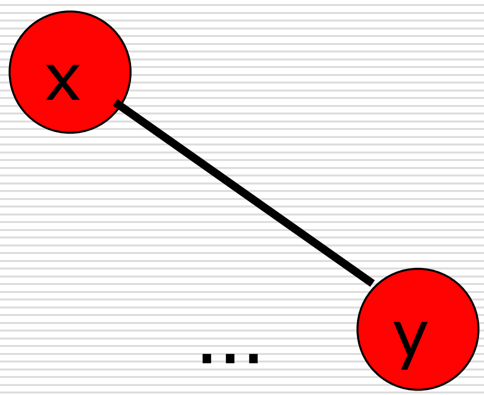
Elimination order influence

- {f(x), f(x), f(x,y)}
- Order: y, x



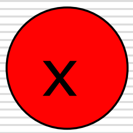
Elimination order influence

- {f(x), f(x), f(x,y)}
- Order: y, x



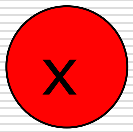
Elimination order influence

- {f(x), f(x), f(x)}
- Order: x



Elimination order influence

- {f(x), f(x), f(x)}
- Order: x

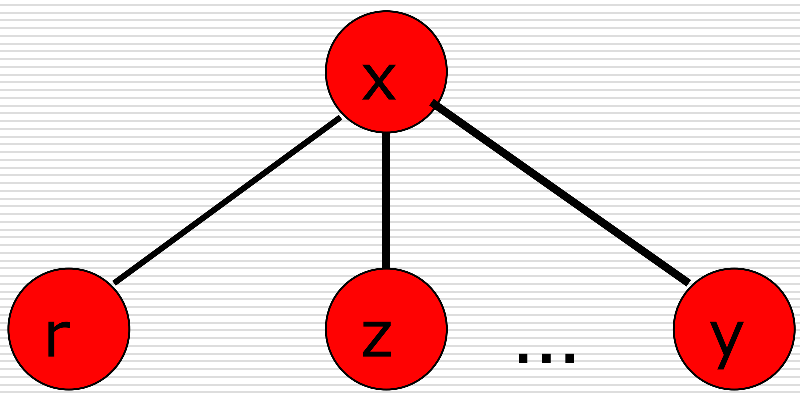


Elimination order influence

- $\{f()\}$
- Order:

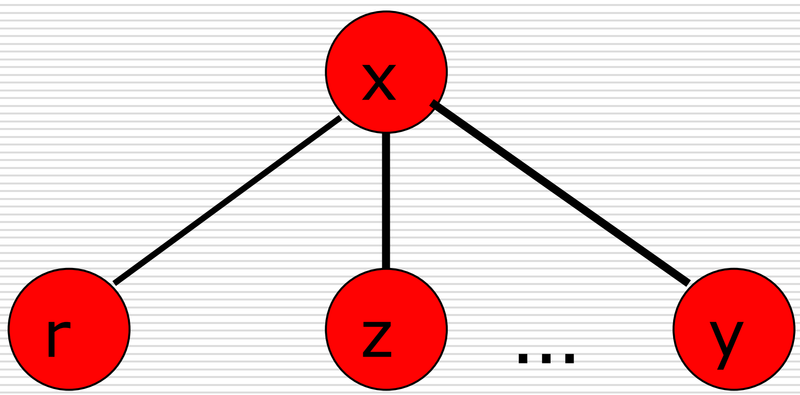
Elimination order influence

- {f(x,r), f(x,z), …, f(x,y)}
- Order: x, y, z, …, r



Elimination order influence

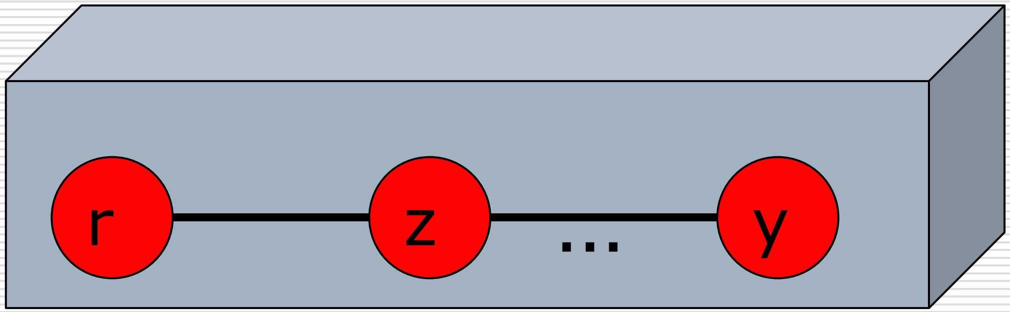
- {f(x,r), f(x,z), …, f(x,y)}
- Order: x, y, z, …, r



Elimination order influence

- {f(r,z,…,y)}
- Order: y, z, r

CLIQUE



Induced width

- For $G=(V,E)$ and a given elimination (vertex) ordering, the largest degree encountered is the **induced width** of the ordered graph

- Minimizing induced width is NP-hard.

History / terminology

- SAT: Directed Resolution (Davis and Putnam, 60)
- Operations Research: Non serial **dynamic programming** (Bertelé Brioschi, 72)
- Databases: Acyclic DB (Beeri et al 1983)
- Bayesian nets: Join-tree (Pearl 88, Lauritzen et Spiegelhalter 88)
- Constraint nets: Adaptive Consistency (Dechter and Pearl 88)

Boosting search with variable elimination: BB-VE(k)

- At each node
  - **Select an unassigned variable x_i**
  - If **$deg_i \leq k$** then **eliminate x_i**
  - Else **branch on the values of x_i**

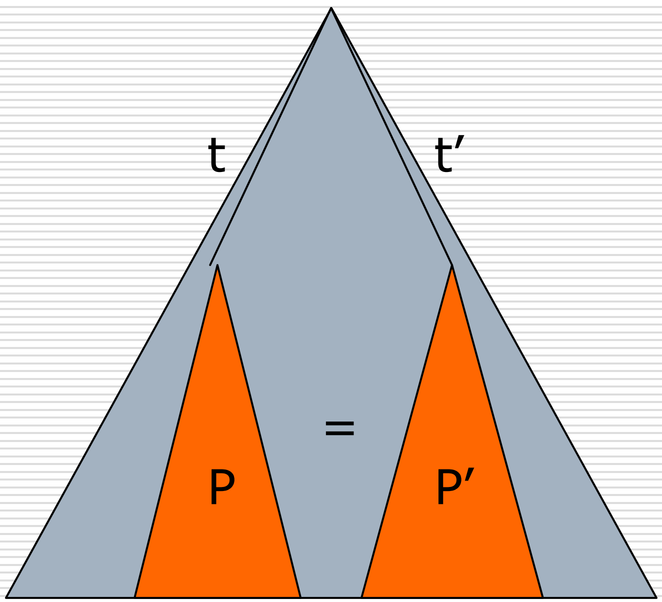
- Properties
  - BE-VE(-1) is BB
  - BE-VE(w*) is VE
  - BE-VE(1) is similar to cycle-cutset

Boosting search with variable elimination

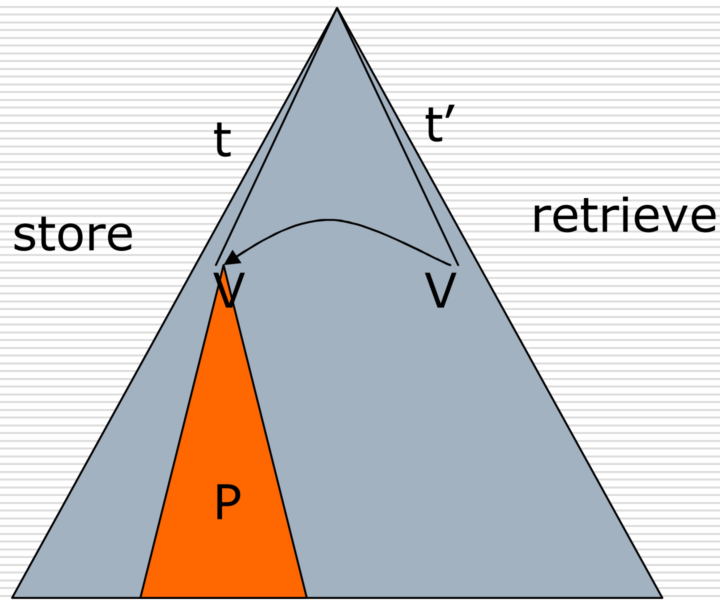
- Ex: still-life (academic problem)
  - Instance: n=14
    - #var:196 , #val:2
  - Ilog Solver → 5 days
  - Variable Elimination → 1 day
  - BB-VE(18) → 2 seconds

Memoization fights thrashing

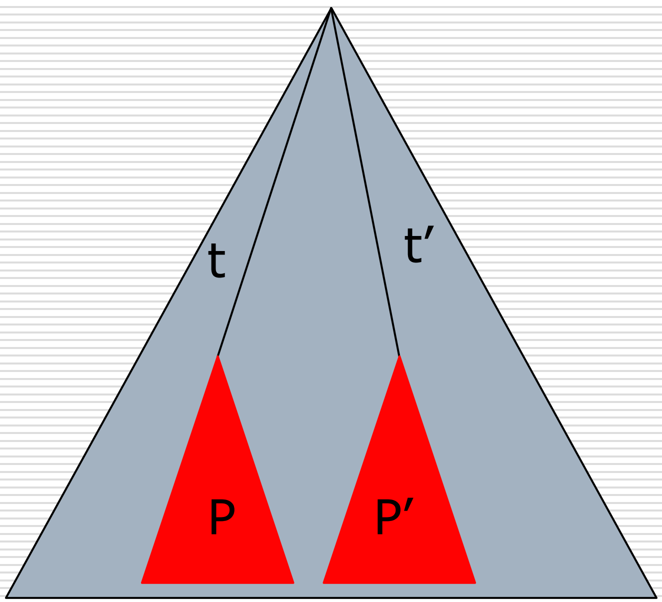
Different nodes,
Same subproblem



Detecting subproblems equivalence is hard



Context-based memoization



- P=P′, if
  - |t|=|t′| and
  - same assign. to partially assigned cost functions

Memoization

- Depth-first B&B with,
  - context-based memoization
  - independent sub-problem detection
- ... is essentialy equivalent to VE
  - Therefore space expensive
- Fresh approach: Easier to incorporate typical tricks such as propagation, symmetry breaking,...
- Algorithms:
  - Recursive Cond. (Darwiche 2001)
  - BTD (Jégou and Terrioux 2003)
  - AND/OR (Dechter et al, 2004)

Adaptive memoization: time/space tradeoff

SAT inference

- In SAT, inference = resolution

$$\begin{array}{c} x \vee A \\ \neg x \vee B \\ \hline A \vee B \end{array}$$

- Effect: transforms explicit knowledge into implicit
- Complete inference:
  - Resolve until quiescence
  - Smart policy: variable by variable (Davis & Putnam, 60). Exponential on the induced width.

Fair SAT Inference

$$(x \vee A, u), (\neg x \vee B, w) \quad \text{Ł} \quad \begin{cases} (A \vee B, m), \\ (x \vee A, u \ominus m), \\ (\neg x \vee B,\ w \ominus m), \\ (x \vee A \vee \neg B, m), \\ (\neg x \vee \neg A \vee B, m) \end{cases}$$

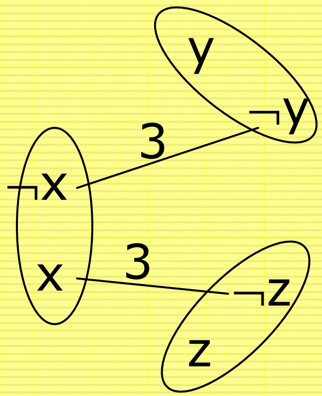
where:

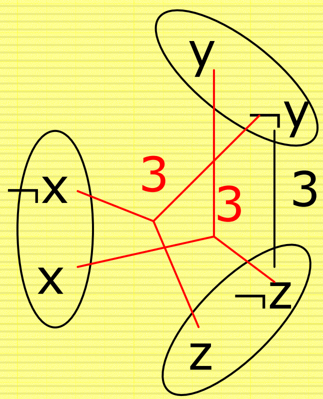
$$m = \min\{u, w\}$$

- <u>Effect</u>: *moves* knowledge

Example: Max-SAT ($\oplus$=+, $\ominus$=-)

$$\left.\begin{matrix}(x \lor y, 3), \\ (\neg x \lor z, 3)\end{matrix}\right\} = \left\{\begin{matrix}(y \lor z, 3), \\ (x \lor y, 3-3), \\ (\neg x \lor z, 3-3), \\ (x \lor y \lor \neg z, 3), \\ (\neg x \lor \neg y \lor z, 3)\end{matrix}\right.$$

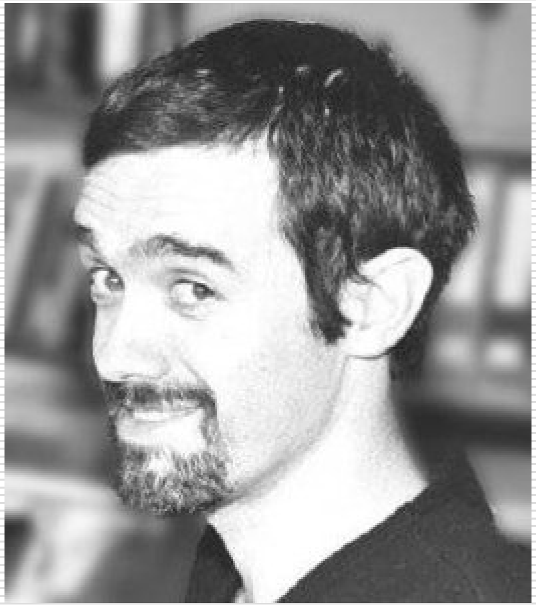




Properties (Max-SAT)

- In SAT, collapses to classical resolution
- Sound and complete
- Variable elimination:
  - Select a variable x
  - Resolve on x until quiescence
  - Remove all clauses mentioning x
- Time and space complexity: exponential on the *induced width*

Change



Incomplete inference

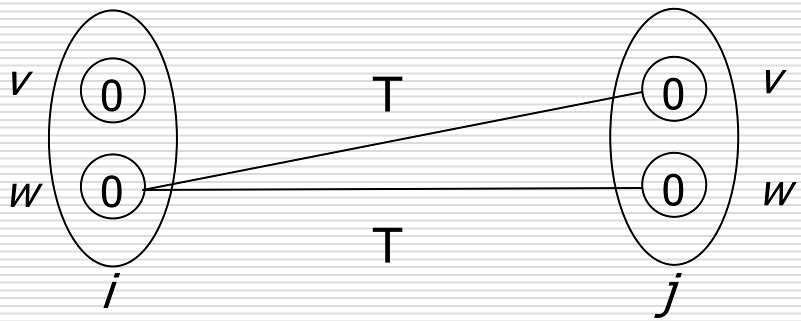
- Local consistency
- Restricted resolution

Incomplete inference

- Tries to trade completeness for space/time
  - Produces only specific classes of cost functions
  - Usually in polynomial time/space
- Local consistency: node, arc…
  - Equivalent problem
  - Compositional: transparent use
  - Provides a lb on optimal cost consistency

Classical arc consistency

- A CSP is AC iff for any x_i and c_{ij}
  - $c_i = c_i \bowtie (c_{ij} \bowtie c_j)[x_i]$
  - namely, $(c_{ij} \bowtie c_j)[x_i]$ brings no new information on x_i



$c_{ij} \bowtie c_j$

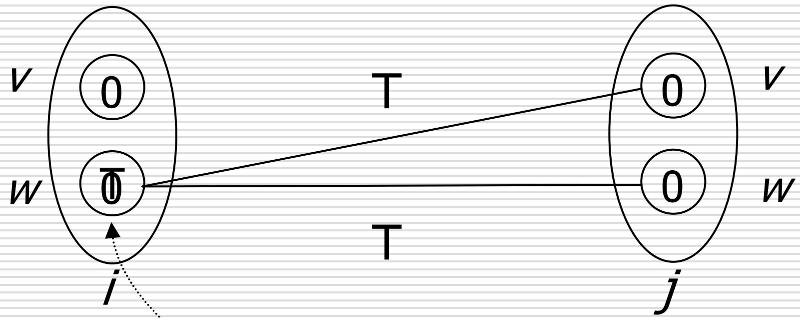
x_i	x_j	c_{ij}
v	v	0
v	w	0
w	v	T
w	w	T

$(c_{ij} \bowtie c_j)[x_i]$

x_i	$c(x_i)$
v	0
w	T

Enforcing AC

- for any x_i and c_{ij}
  - $c_i := c_i \bowtie (c_{ij} \bowtie c_j)[x_i]$ until fixpoint (unique)



$c_{ij} \bowtie c_j$

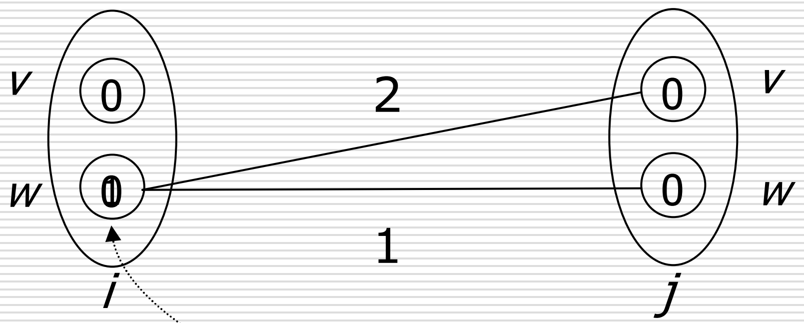
x_i	x_j	c_{ij}
v	v	0
v	w	0
w	v	T
w	w	T

$(c_{ij} \bowtie c_j)[x_i]$

x_i	$c(x_i)$
v	0
w	T

Arc consistency and soft constraints

- for any x_i and f_{ij}
  - $f=(f_{ij} \oplus f_j)[x_i]$ brings no new information on x_i



$f_{ij} \oplus f_j$

x_i	x_j	f_{ij}
v	v	0
v	w	0
w	v	2
w	w	1

$(f_{ij} \oplus f_j)[x_i]$

X_i	$f(x_i)$
v	0
w	1

Always equivalent iff $\oplus$ idempotent

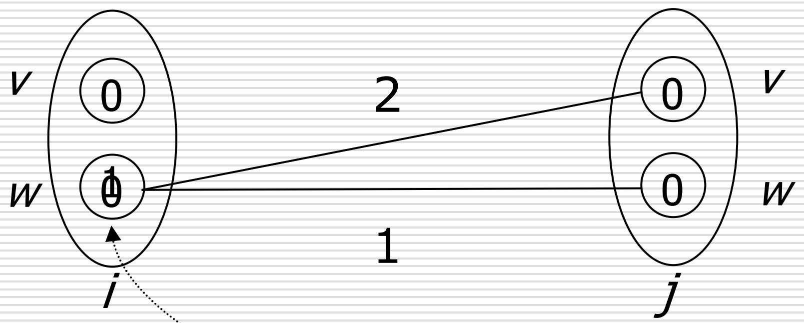
Idempotent soft CN

- The previous operational extension works on any idempotent semiring CN
  - Chaotic iteration of local enforcing rules until fixpoint
  - Terminates and yields an equivalent problem
  - Extends to generalized k-consistency

  - Total order: idempotent ($\oplus$ = max)

Non idempotent: weighted CN

- for any x_i and f_{ij}
  - $f=(f_{ij} \oplus f_j)[x_i]$ brings no new information on x_i



$f_{ij} \oplus f_j$

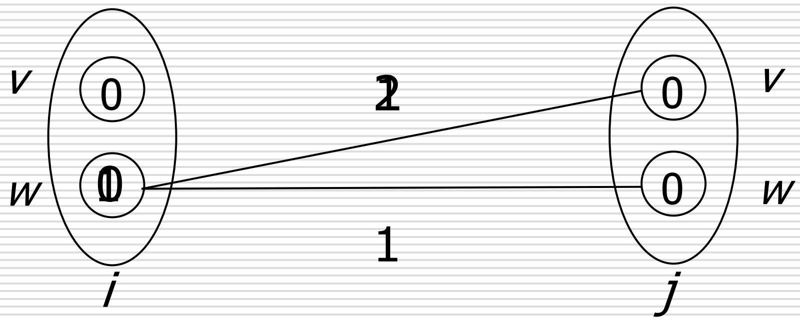
x_i	x_j	f_{ij}
v	v	0
v	w	0
w	v	2
w	w	1

$f_{ij} \oplus f_j\,[x_i]$

x_i	$f(x_i)$
v	0
w	1

EQUIVALENCE LOST

IV - Subtraction of cost functions (fair)



- Combination+Subtraction: equivalence preserving transformation

(K,Y) **equivalence preserving inference**

- For a set K of cost functions and a scope Y
  - Replace K by $(\oplus K)$
  - Add $(\oplus K)[Y]$ to the CN (implied by $\oplus K$)
  - Subtract $(\oplus K)[Y]$ from $(\oplus K)$
- Yields an equivalent network
- All implicit information on **Y** in **K** is explicit
- Repeat for a class of (K,Y) until fixpoint

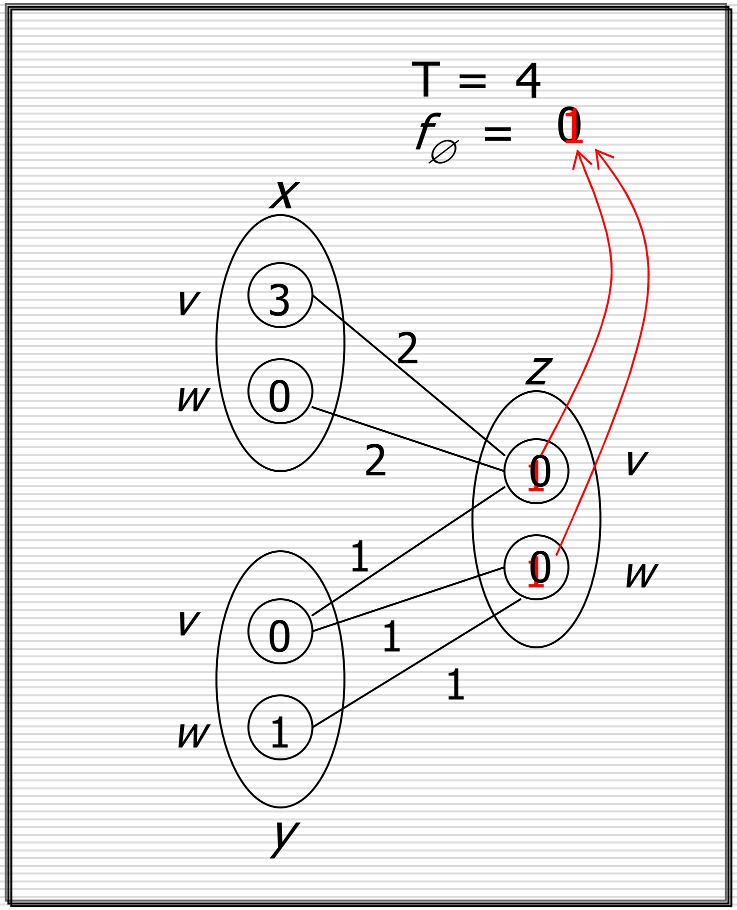
Node Consistency (NC*): ({$f_\varnothing$,f_i}, $\varnothing$) EPI

- For any variable X_i
  - $\forall a, f_\varnothing + f_i(a) < T$
  - $\exists\, a, f_i(a) = 0$

Or T may decrease:
back-propagation

- Complexity:

O(nd)

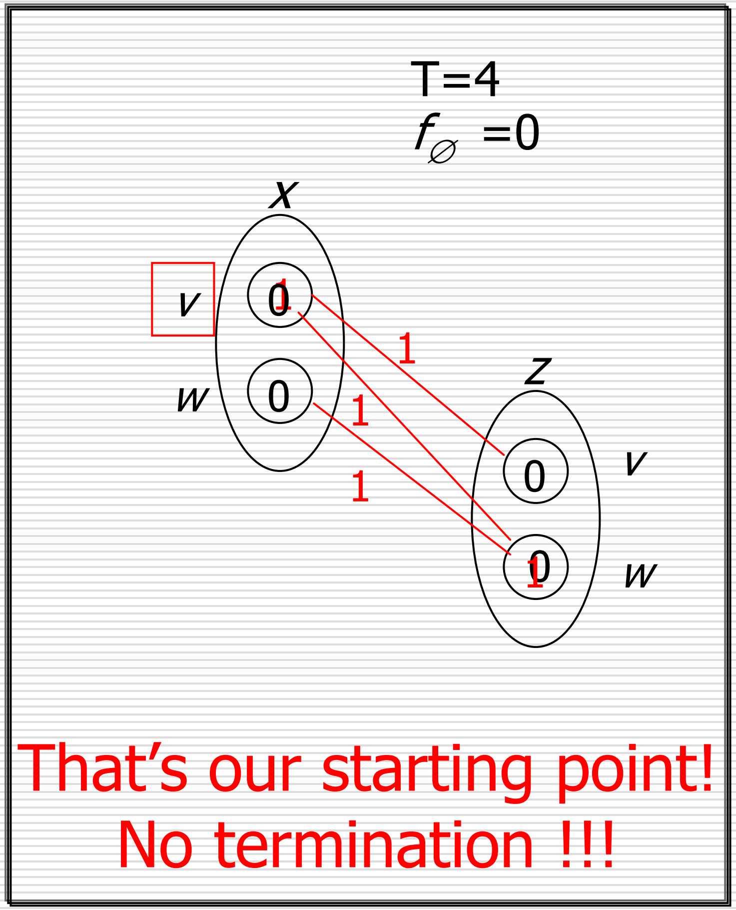


Full AC (FAC*): $(\{f_{ij},f_j\},\{x_i\})$ EPI

- NC*
- For all f_{ij}
  - $\forall a \exists b$

    $f_{ij}(a,b) + f_j(b) = 0$

    (full support)

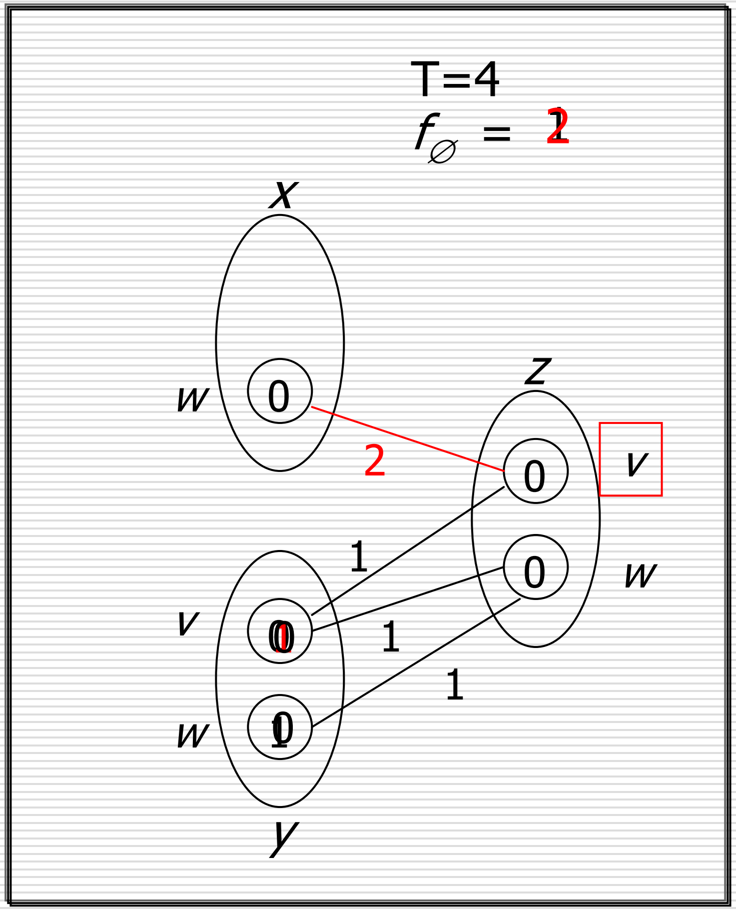


Arc Consistency (AC*): $(\{f_{ij}\},\{x_i\})$ EPI

- NC*
- For all f_{ij}
  - $\forall a \exists b$

    $f_{ij}(a,b) = 0$
- b is a *support*
- complexity:

  $\mathbf{O}(n^2 d^3)$



Neighborhood Resolution

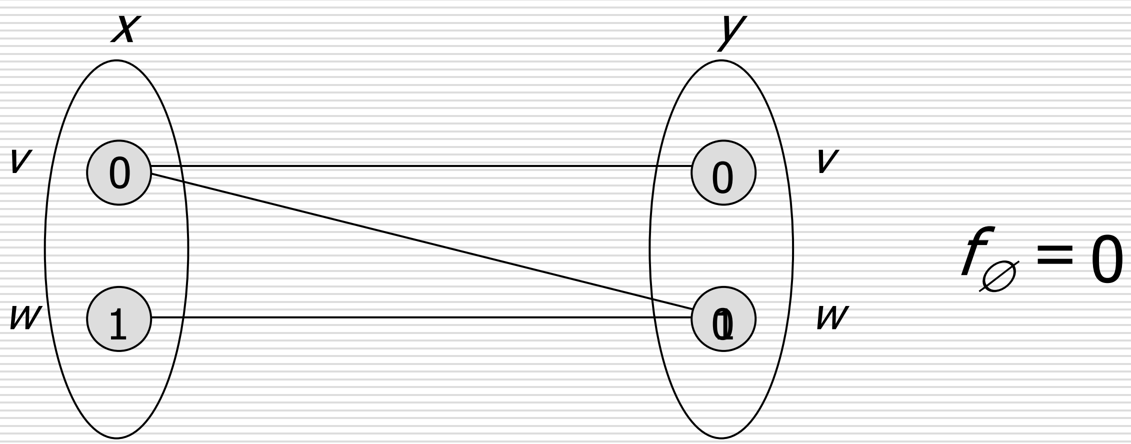
$$(x \vee \boldsymbol{A},u), (\neg x \vee \boldsymbol{A},w) \quad \text{Ł} \quad \begin{cases} (A,m), \\ (x \vee A, u \ominus m), \\ (\neg x \vee A,\ w \ominus m), \\ (x \vee A \vee \neg A, m), \\ (\neg x \vee \neg A \vee A, m) \end{cases}$$

- if |A|=0, enforces node consistency
- if |A|=1, enforces arc consistency

Confluence is lost

$f_{\varnothing} = 0$

Confluence is lost



Finding an AC closure that maximizes the lb is an NP-hard problem (Cooper & Schiex 2004).

Well… one can do better in pol. time (OSAC, IJCAI 2007)

Hierarchy

Special case: CSP (Top=1)

NC* $\mathbf{O}(nd)$ **NC**

AC* $\mathbf{O}(n^2d^3)$

DAC* $\mathbf{O}(ed^2)$ **DAC**

FDAC* $\mathbf{O}(end^3)$ **AC**

EDAC* $\mathbf{O}(ed^2 \max\{nd, T\})$

Boosting search with LC

BT(X,D,C)

    if (X=∅) then Top :=$f_{\varnothing}$

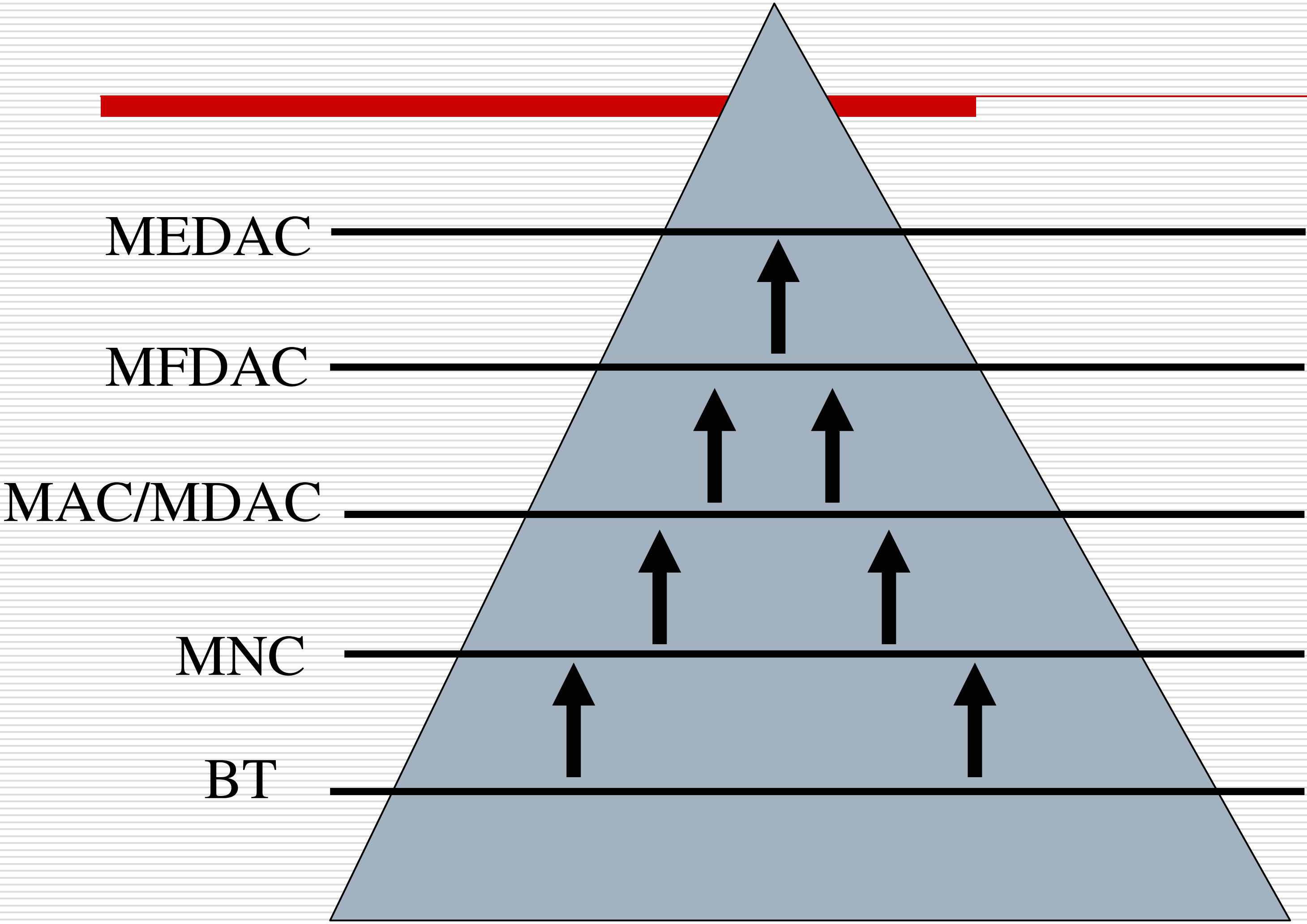
    else

        x_j := selectVar(X)

        forall $a \in D_j$ do

            $\forall_{f_S \in C \text{ s.t. } x_j \in S}$ f_S := f_S[x_j =a]

            if (**LC**) then BT(X-{x_j},D-{D_j},C)


MEDAC
MFDAC
MAC/MDAC
MNC
BT

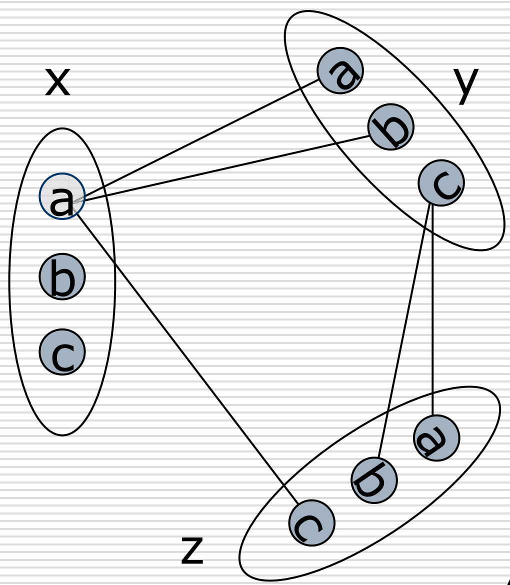
Boosting Systematic Search with Local consistency

Frequency assignment problem

- CELAR6-sub4 (22 var, 44 val, 477 cost func):
  - MNC*Ł 1 year
  - MFDAC* Ł 1 hour
- CELAR6 (100 var, 44 val, 1322 cost func):
  - MEDAC+memoization Ł 3 hours (toolbar-BTD)

Beyond Arc Consistency

- Path inverse consistency PIC (Debryune & Bessière)



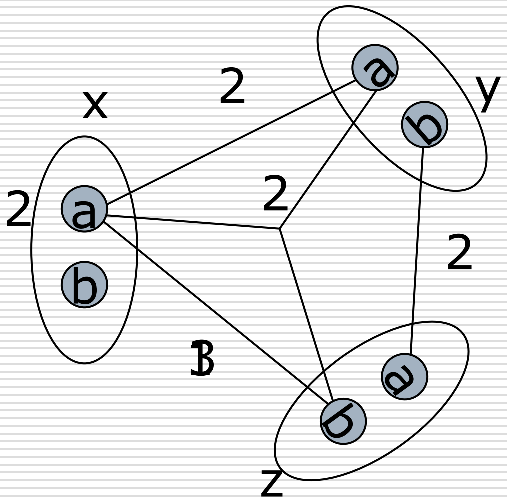
(x,a) can be pruned because there are two other variables y,z such that (x,a) cannot be extended to any of their values.

$(\{f_y, f_z, f_{xy}, f_{xz}, f_{yz}\},\{x\})$ EPI

Beyond Arc Consistency

- Soft Path inverse consistency PIC*

$(\{f_y, f_z, f_{xy}, f_{xz}, f_{yz}\},x)$ EPI



$f_y \oplus f_z \oplus f_{xy} \oplus f_{xz} \oplus f_{yz}$

x	y	z	
a	a	a	0
a	a	b	3
a	b	a	0
a	b	b	1
b	a	a	0
b	a	b	0
b	b	a	2
b	b	b	0

$(f_y \oplus f_z \oplus f_{xy} \oplus f_{xz} \oplus f_{yz})[x]$

x	
a	2
b	0

Hyper-resolution (2 steps)

$$\left.\begin{array}{l}(l \vee h \vee A, u),\\ (\neg l \vee q \vee A, v),\\ (\neg h \vee q \vee A, u)\end{array}\right\} = \begin{array}{l}(h \vee q \vee A, m),\\ \left.\begin{array}{l}(l \vee h \vee A, u-m),\\ (\neg l \vee q \vee A, v-m),\\ (l \vee h \vee \neg q \vee A, m),\\ (\neg l \vee q \vee \neg h \vee A, m),\end{array}\right\} = \\ (\neg h \vee q \vee A, u)\end{array} \begin{array}{l}(q \vee A, m'),\\ (h \vee q \vee A, m-m'),\\ (\neg h \vee q \vee A, u-m'),\\ (l \vee h \vee A, u-m),\\ (\neg l \vee q \vee A, v-m),\\ (l \vee h \vee \neg q \vee A, m),\\ (\neg l \vee q \vee \neg h \vee A, m)\end{array}$$

if |A|=0, equal to soft PIC

Impressive empirical speed-ups

Complexity & Polynomial classes

Tree = induced width 1

Idempotent $\oplus$ or not…

Polynomial classes
Idempotent VCSP: min-max CN

- Can use α-cuts for lifting CSP classes
  - Sufficient condition: the polynomial class is «conserved» by α-cuts

  - Simple TCSP are TCSP where all constraints use 1 interval: $x_i - x_j \in [a_{ij}, b_{ij}]$

  - Fuzzy STCN: any slice of a cost function is an interval (semi-convex function) (Rossi et al.)

Hardness in the additive case
(weighted/boolean)

- MaxSat is MAXSNP complete (no PTAS)
  - Weighted MaxSAT is FP^{NP}-complete
  - MaxSAT is $FP^{NP[O(\log(n))]}$ complete: weights !
  - MaxSAT tractable langages fully characterized (Creignou 2001)

- MaxCSP langage: $f_{eq}(x,y) : (x = y)\ ?\ 0 : 1$ is NP-hard.
  - Submodular cost function lang. is polynomial.
    $(u \leq x,\ v \leq y \quad f(u,v)+f(x,y) \leq f(u,y)+f(x,v))$ (Cohen et al.)

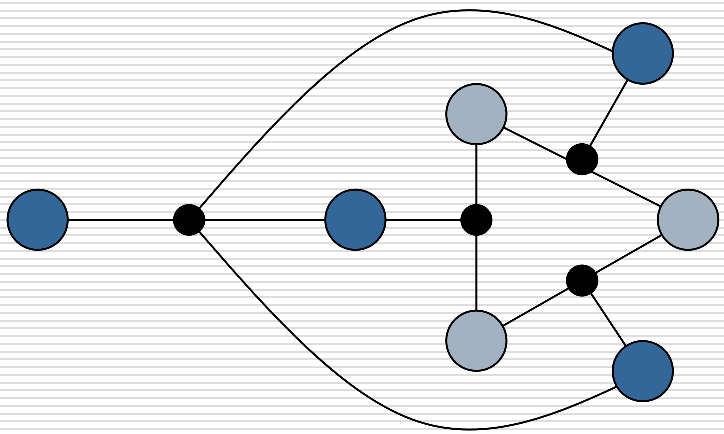
Integration of soft constraints into classical constraint programming

Soft as hard

Soft local consistency as a global constraint

Soft constraints as hard constraints

- one extra variable x_s per cost function f_S
- all with domain E
- $f_S \rightarrow c_{S \cup \{x_S\}}$ allowing $(t, f_S(t))$ for all $t \in \ell(S)$
- one variable $x_C = \oplus\, x_s$ (global constraint)



Soft as Hard (SaH)

- Criterion represented as a variable
- Multiple criteria = multiple variables
- Constraints on/between criteria

- Weaknesses:
  - Extra variables (domains), increased arities
  - SaH constraints give weak GAC propagation
  - Problem structure changed/hidden

$\geq$

Soft AC « stronger than » SasH GAC

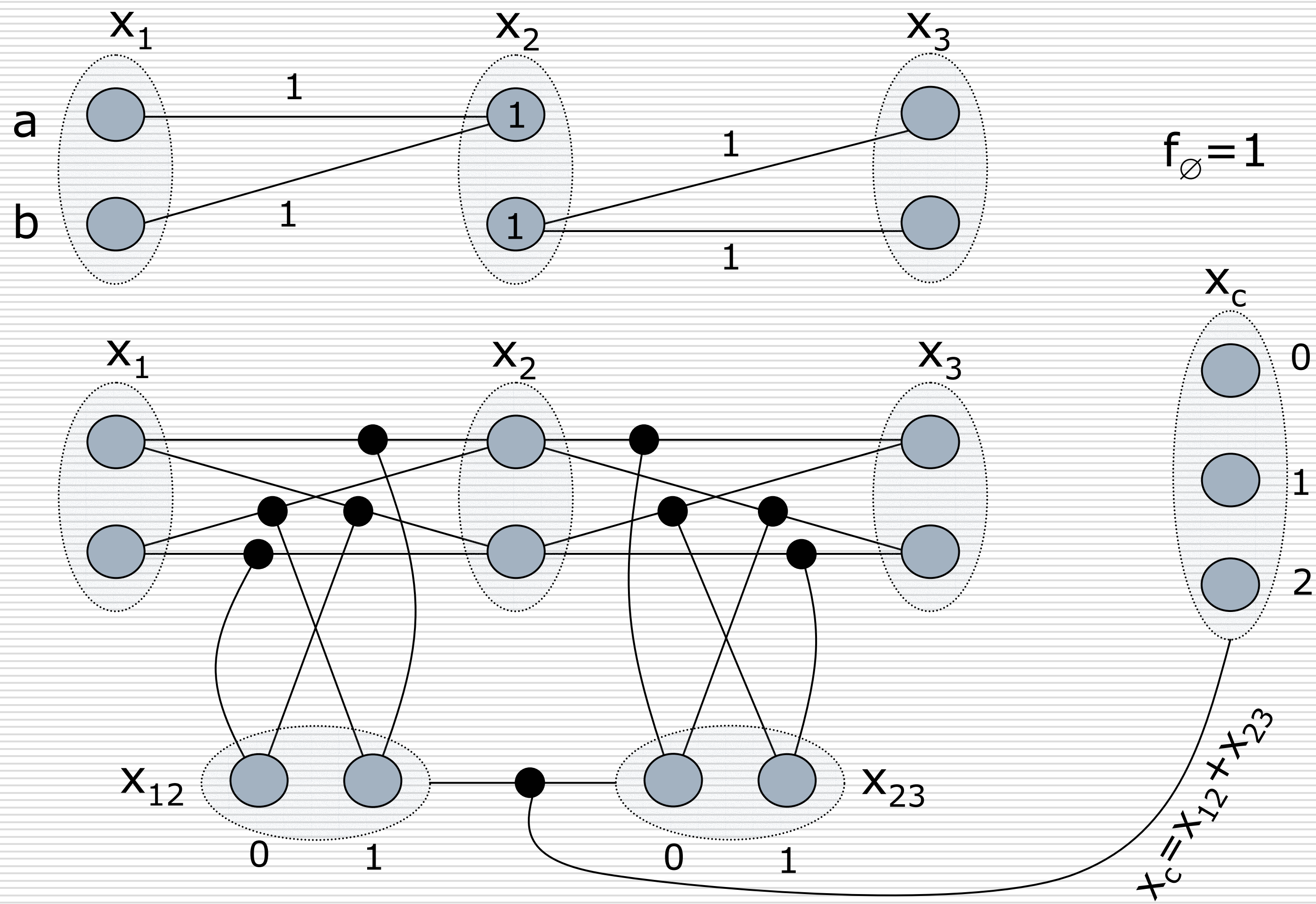
- Take a WCSP
- Enforce Soft AC on it

  Each cost function contains at least one tuple with a 0 cost (definition)
- Soft as Hard: the cost variable x_C will have a lb of 0

- The lower bound cannot improve by GAC

>

Soft AC « stronger than » SasH GAC



Soft local Consistency as a Global constraint ($\oplus$=+)

- Global constraint: Soft(X,F,C)
  - X variables
  - F cost functions
  - C interval cost variable (ub = T)

- Semantics: X U{C} satisfy Soft(X,F,C) iff

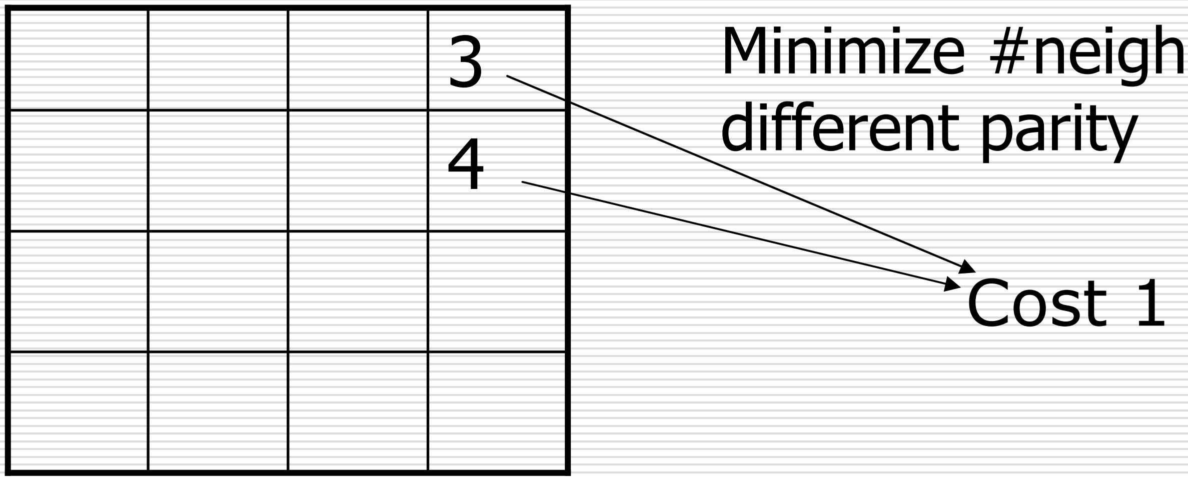
$$\sum f(X)=C$$

- Enforcing GAC on Soft is NP-hard
- Soft consistency: filtering algorithm ($lb \geq f_{\varnothing}$)

Ex: Spot 5 (Earth satellite sched.)

- For each requested photography:
  - € lost if not taken , Mb of memory if taken
- <u>variables</u>: requested photographies
- <u>domains</u>: {0,1,2,3}
- <u>constraints</u>:
  - $\{r_{ij}, r_{ijk}\}$ binary and ternary hard costraints
  - Sum(X)<Cap. global memory bound
  - Soft(X,F_1,€) bound € loss

Example: soft quasi-group (motivated by sports scheduling)



Minimize #neighbors of different parity

- Alldiff($x_{i1},\dots,x_{in}$) i=1..m
- Alldiff($x_{1j},\dots,x_{mj}$) j=1..n
- Soft(X,{f_{ij}},[0..k],+)

Global soft constraints

Global soft constraints

- Idea: define a library of useful but non-standard *objective functions* along with efficient filtering algorithms
  - AllDiff (2 semantics: Petit et al 2001, van Hoeve 2004)
  - Soft global cardinality (van Hoeve et al. 2004)
  - Soft regular (van Hoeve et al. 2004)
  - … all enforce reified GAC

Conclusion

- A large subset of classic CN body of knowledge has been extended to soft CN, efficient solving tools exist.
- Much remains to be done:
  - Extension: to other problems than optimization (counting, quantification…)
  - Techniques: symmetries, learning, knowledge compilation…
  - Algorithmic: still better lb, other local consistencies or dominance. Global (SoftAsSoft). Exploiting problem structure.
  - Implementation: better integration with classic CN solver (Choco, Solver, Minion…)
  - Applications: problem modelling, solving, heuristic guidance, partial solving.

30” of publicity J

Open source libraries
Toolbar and Toulbar2

- Accessible from the **Soft wiki site**:

  carlit.toulouse.inra.fr/cgi-bin/awki.cgi/SoftCSP
- Alg: *BE-VE,MNC,MAC,MDAC,MFDAC,MEDAC,MPIC,BTD*
- ILOG connection, large domains/problems…
- Read MaxCSP/SAT (weighted or not) and ERGO format
- Thousands of benchmarks in standardized format
- Pointers to **other solvers** (MaxSAT/CSP) **Pwd: bia31**
- Forge mulcyber.toulouse.inra.fr/projects/toolbar (toulbar2)

Thank you for your attention
This is it !

- S. Bistarelli, U. Montanari and F. Rossi, Semiring-based Constraint Satisfaction and Optimization, *Journal of ACM*, vol.44, n.2, pp. 201-236, March 1997.
- S. Bistarelli, H. Fargier, U. Montanari, F. Rossi, T. Schiex, G. Verfaillie. Semiring-Based CSPs and Valued CSPs: Frameworks, Properties, and Comparison. *CONSTRAINTS*, Vol.4, N.3, September 1999.
- S. Bistarelli, R. Gennari, F. Rossi. Constraint Propagation for Soft Constraint Satisfaction Problems: Generalization and Termination Conditions , in *Proc. CP 2000*
- C. Blum and A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, 35(3):268-308, 2003.
- *T. Schiex,* Arc consistency for soft constraints*, in Proc. CP'2000.*
- *M. Cooper, T. Schiex.* Arc consistency for soft constraints, *Artificial Intelligence, Volume 154 (1-2), 199-227* 2004.
- *M. Cooper*. Reduction Operations in fuzzy or valued constraint satisfaction problems. Fuzzy Sets and Systems 134 (3) 2003.
- *A. Darwiche.* Recursive Conditioning**.** Artificial Intelligence. Vol 125, No 1-2, pages 5-41.
- R. Dechter. Bucket Elimination: A unifying framework for Reasoning**.** *Artificial Intelligence, October, 1999.*
- R. Dechter, Mini-Buckets: A General Scheme For Generating Approximations In Automated Reasoning *In Proc. Of IJCAI97*

References

- *S. de Givry, F. Heras, J. Larrosa & M. Zytnicki. Existential arc consistency: getting closer to full arc consistency in weighted CSPs. In IJCAI 2005.*
- *W.-J. van Hoeve, G. Pesant and L.-M. Rousseau. On Global Warming: Flow-Based Soft Global Constraints. Journal of Heuristics 12(4-5), pp. 347-373, 2006.*
- P. Jegou & C. Terrioux. Hybrid backtracking bounded by tree-decomposition of constraint networks. Artif. Intell. 146(1): 43-75 (2003)
- J. Larrosa & T. Schiex. Solving Weighted CSP by Maintaining Arc Consistency. *Artificial Intelligence*. 159 (1-2): 1-26, 2004.
- *J. Larrosa and T. Schiex.* In the quest of the best form of local consistency for Weighted CSP, Proc. of IJCAI'03
- J. Larrosa, P. Meseguer, T. Schiex Maintaining Reversible DAC for MAX-CSP. *Artificial Intelligence*.107(1), pp. 149-163.
- R. Marinescu and R. Dechter. AND/OR Branch-and-Bound for Graphical Models. *In proceedings of IJCAI'2005.*
- J.C. Regin, T. Petit, C. Bessiere and J.F. Puget. An original constraint based approach for solving over constrained problems. In Proc. CP'2000.
- *T. Schiex, H. Fargier et G. Verfaillie.* Valued Constraint Satisfaction Problems: hard and easy problems *In* Proc. of IJCAI 95.
- *G. Verfaillie, M. Lemaitre et T. Schiex.* Russian Doll Search Proc. of AAAI'96.

References

- M. Bonet, J. Levy and F. Manya. A complete calculus for max-sat. In SAT 2006.
- M. Davis & H. Putnam. A computation procedure for quantification theory. In JACM 3 (7) 1960.
- I. Rish and R. Dechter. Resolution versus Search: Two Strategies for SAT. *In Journal of Automated Reasoning,* 24 (1-2), 2000.
- F. Heras & J. Larrosa. New Inference Rules for Efficient Max-SAT Solving. In *AAAI 2006*.
- J. Larrosa, F. Heras. Resolution in Max-SAT and its relation to local consistency in weighted CSPs. *In IJCAI 2005*.
- C.M. Li, F. Manya and J. Planes. Improved branch and bound algorithms for max-sat. In AAAI 2006.
- H. Shen and H. Zhang. Study of lower bounds for max-2-sat. In proc. of AAAI 2004.
- Z. Xing and W. Zhang. MaxSolver: An efficient exact algorithm for (weighted) maximum satisfiability. Artificial Intelligence 164 (1-2) 2005.

SoftasHard GAC vs. EDAC 25 variables, 2 values binary MaxCSP

- Toolbar MEDAC
  - opt=34
  - 220 nodes
  - cpu-time = 0″

- GAC on SoftasHard, ILOG Solver 6.0, solve
  - opt = 34
  - 339136 choice points
  - cpu-time: 29.1″
  - Uses table constraints

Other hints on SoftasHard GAC

- MaxSAT as Pseudo Boolean $\Leftrightarrow$ SoftAsHard
  - For each clause:

    $c = (x \vee \ldots \vee z, p_c) \quad c_{SAH} = (x \vee \ldots \vee z \vee r_c)$
  - Extra cardinality constraint:

    $$\Sigma\, p_c.r_c \leq k$$
  - Used by SAT4JMaxSat (MaxSAT competition).

MaxSAT competition (SAT 2006) Unweighted MaxSAT

Set Name	#Instances	MaxSatz	Toolbar	Lazy	ChaffBS	ChaffLS	SAT4Jmaxsat
Max-Cut (brock)	12	13.35(12)	57.50(12)	178.48(12)	0.00(0)	0.00(0)	0.00(0)
Max-Cut (c-fat)	7	0.07(5)	21.05(5)	151.13(5)	0.01(2)	0.01(2)	0.85(2)
Max-Cut (hamming)	6	180.12(3)	575.52(3)	42.06(2)	0.00(0)	0.00(0)	0.00(0)
Max-Cut (johnson)	4	45.39(3)	134.68(3)	2.45(2)	0.00(0)	0.00(0)	0.00(0)
Max-Cut (keller)	2	6.12(2)	17.25(2)	69.86(2)	0.00(0)	0.00(0)	0.00(0)
Max-Cut (p_hat)	12	15.84(12)	61.86(12)	192.05(12)	0.00(0)	0.00(0)	0.00(0)
Max-Cut (san)	11	275.05(11)	65.02(7)	249.83(7)	0.00(0)	0.00(0)	0.00(0)
Max-Cut (sanr)	4	71.98(4)	266.86(4)	80.78(3)	0.00(0)	0.00(0)	0.00(0)
Max-Cut (random)	40	5.58(40)	34.67(40)	752.34(25)	0.00(0)	0.00(0)	0.00(0)
Max-Cut (spinglass)	5	44.92(3)	4.96(2)	48.21(2)	9.97(1)	6.19(1)	0.00(0)
Max-One	45	0.02(45)	5.44(45)	81.34(40)	1.00(45)	0.20(45)	2.31(41)
Ramsey	48	8.99(34)	53.14(33)	81.70(28)	53.39(34)	7.36(33)	2.86(32)
Max-2-SAT (60 vars)	50	0.03(50)	0.62(50)	3.27(50)	13.74(10)	25.69(10)	0.00(0)
Max-2-SAT (100 vars)	50	1.40(50)	17.57(50)	235.83(31)	0.70(10)	1.08(10)	24.37(10)
Max-2-SAT (140 vars)	50	7.02(50)	105.61(49)	204.10(23)	272.77(12)	99.86(11)	47.26(11)
Max-2-SAT (discarded)	180	16.79(180)	99.34(175)	141.39(107)	262.04(18)	172.67(14)	59.87(4)
Max-3-SAT (40 vars)	50	1.50(50)	8.09(50)	6.94(50)	0.31(10)	0.28(10)	50.05(11)
Max-3-SAT (60 vars)	50	23.31(50)	264.98(50)	266.70(43)	84.76(11)	68.55(11)	1.96(10)

MaxSAT competition (SAT 2006) Weighted

Set Name	#Instances	Toolbar	Lazy	SAT4Jmaxsat
Auction (paths)	30	249.77(26)	81.24(20)	0.00(0)
Auction (regions)	30	8.16 (30)	2.03 (28)	926.99(6)
Auction (scheduling)	30	132.15(30)	63.33(30)	518.41(8)
Max-Clique (brock)	12	96.76(4)	104.69(4)	0.00(0)
Max-Clique (c-fat)	7	25.19(7)	17.36(7)	346.68(4)
Max-Clique (hamming)	6	134.04(5)	195.05(5)	6.32 (2)
Max-Clique (johnson)	4	53.91(3)	38.64(3)	61.73(2)
Max-Clique (keller)	2	34.12(1)	43.38(1)	0.01 (1)
Max-Clique (mann_a)	4	45.62(3)	0.31 (1)	726.50(2)
Max-Clique (p_hat)	12	325.70(8)	254.14(6)	0.00(0)
Max-Clique (san)	11	25.01(3)	10.88(1)	0.00(0)
Max-Clique (sanr)	4	821.98(3)	790.55(2)	0.00(0)