A Portfolio Approach for Enforcing Minimality in a Tree Decomposition

Daniel J. Geschwender[1,2] R.J. Woodward[1,2] B.Y. Choueiry[1,2] S. D. Scott[2]

[1]Constraint Systems Laboratory
[2]Department of Computer Science and Eng.
University of Nebraska-Lincoln

Acknowledgements
- Experiments conducted at UNL's Holland Computing Center
- Geschwender supported by a NSF Graduate Research Fellowship Grant No. 1041000
- NSF Grants No. RI-111795 and RI-1619344

Daniel Geschwender

- 3rd year PhD student at University of Nebraska – Lincoln's Constraint Systems Laboratory

- Studying high level relational consistencies and automated techniques for determining when to apply them

Constraint Systems Laboratory

- Always ready to play a board game!

Claim: Cluster-level portfolio

We advocate the use of an **algorithm portfolio**

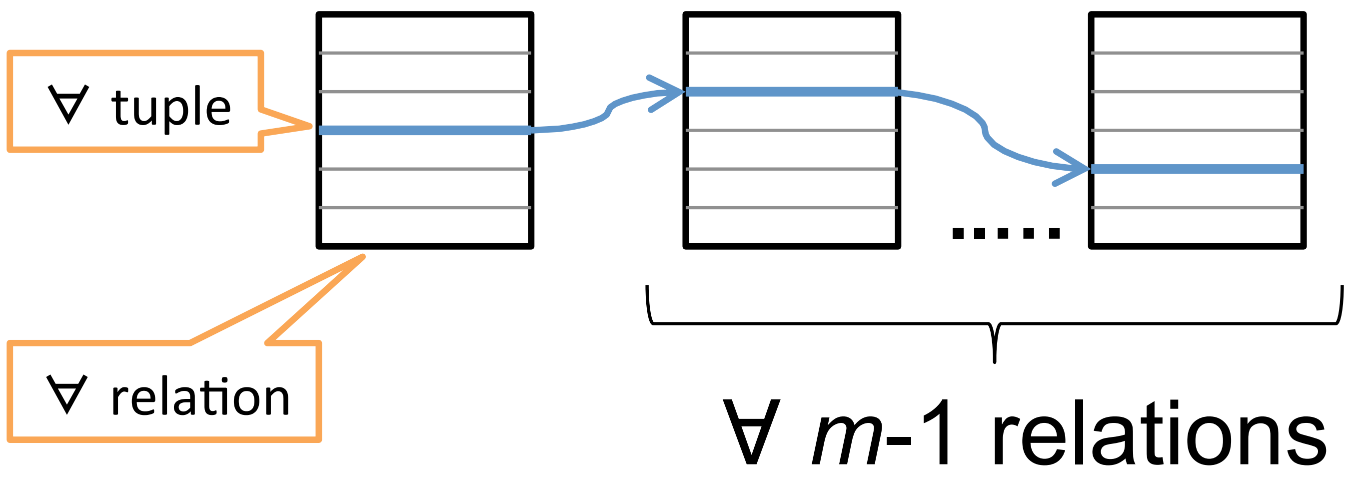
- for enforcing **minimality**
- on the **clusters** of a tree decomposition
- during **lookahead** in a backtrack search for solving CSPs

Outline

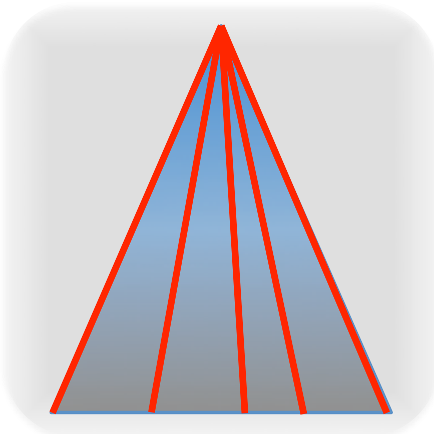
- Background
  - Minimality: property and algorithms (ALLSOL, PERTUPLE)
  - Minimality in a tree decomposition
- Processing clusters: FILTERCLUSTERS
  - GAC interleave
  - Cluster-level portfolio
  - Cluster-processing timeout
- Training the classifier
- Experiments
- Conclusion

Background: Minimality

- Global consistency property
- Every tuple in a relation can be extended to a full solution over the *m* relations



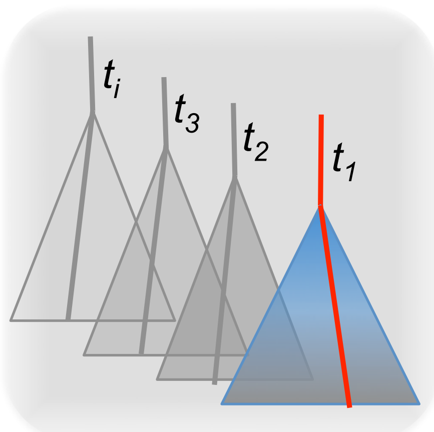
Background: ALLSOL/PERTUPLE



ALLSOL [Karakashian, PhD 2013]

- One search explores the entire search space
- Finds all solutions without storing them, keeps tuples that appear in at least one solution
- Better when there are many 'almost' solutions

Background: AllSol/PerTuple

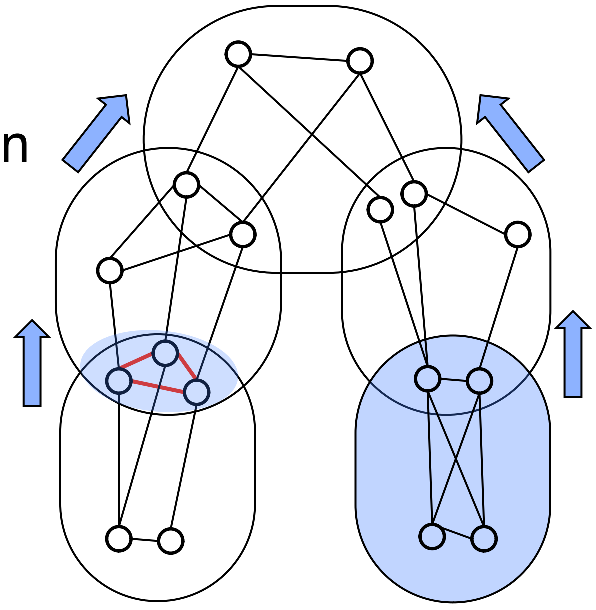


PerTuple [Karakashian, PhD 2013]

- For each tuple, finds one solution where it appears
- Many searches that stop after the first solution
- Better when many solutions are available

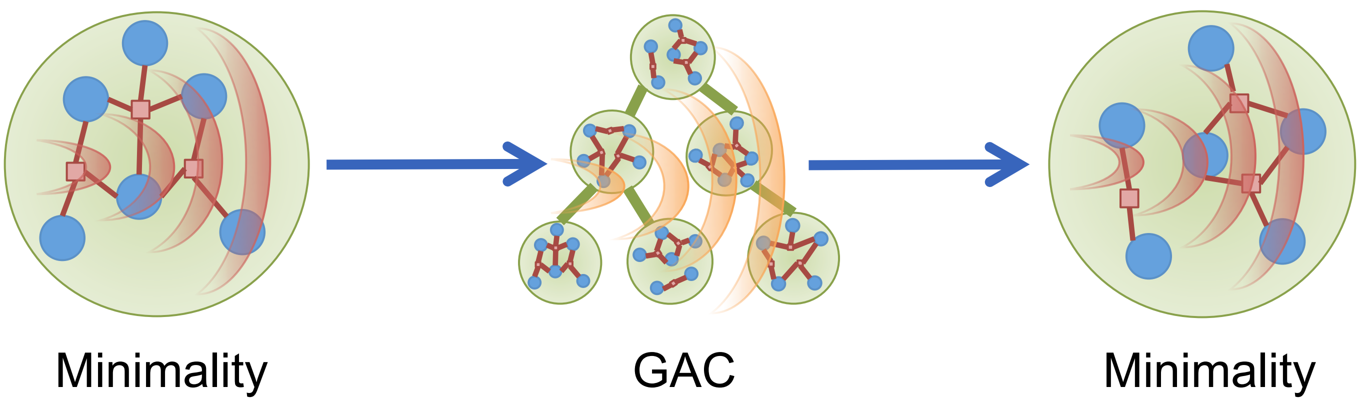
Background: Tree decomposition, minimality

- Minimality on clusters [Karakashian+ AAAI 2013]
  - Build a tree decomposition
  - Localize minimality to clusters
  - During search, after a variable instantiation
    - Enforce minimality on clusters
    - Propagate following tree structure
- FILTERCLUSTERS implements three improvements
  - GAC interleave
  - Cluster-level portfolio
  - Cluster-processing timeout



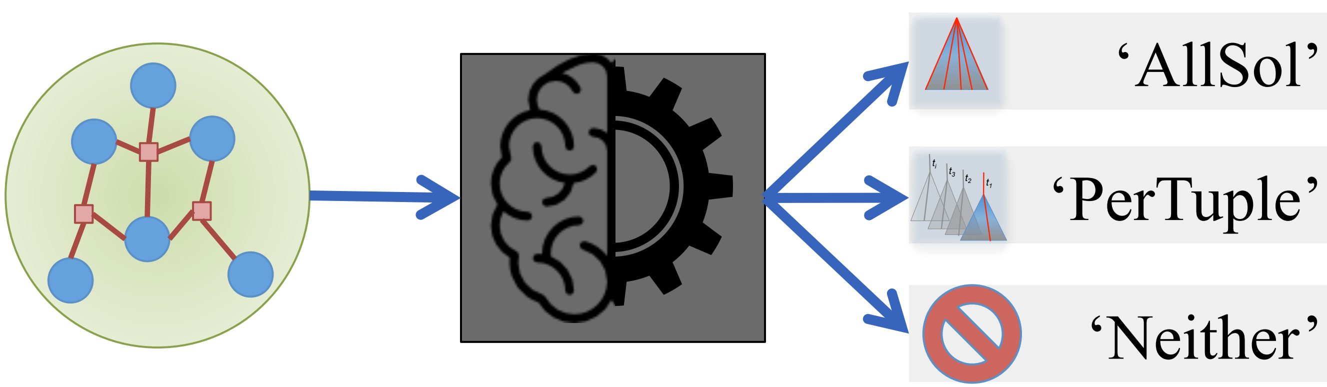
FILTERCLUSTERS: GAC interleave

- It is often beneficial to run a lightweight algorithm (e.g., GAC) prior to running a more costly algorithm
- We extend this idea and interleave a global GAC run between the processing of clusters



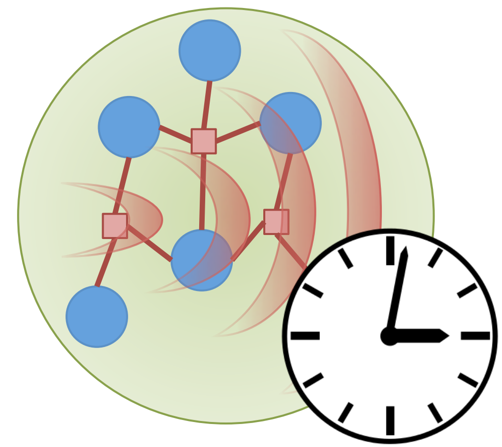
FILTERCLUSTERS: Cluster-level portfolio

- Performance of ALLSOL and PERTUPLE vary
- Sometimes both algorithms are too costly
- Use algorithm portfolio on the cluster level
  - Different algorithms on different clusters
  - Different algorithms on the same cluster during propagation



FilterClusters: Cluster timeout

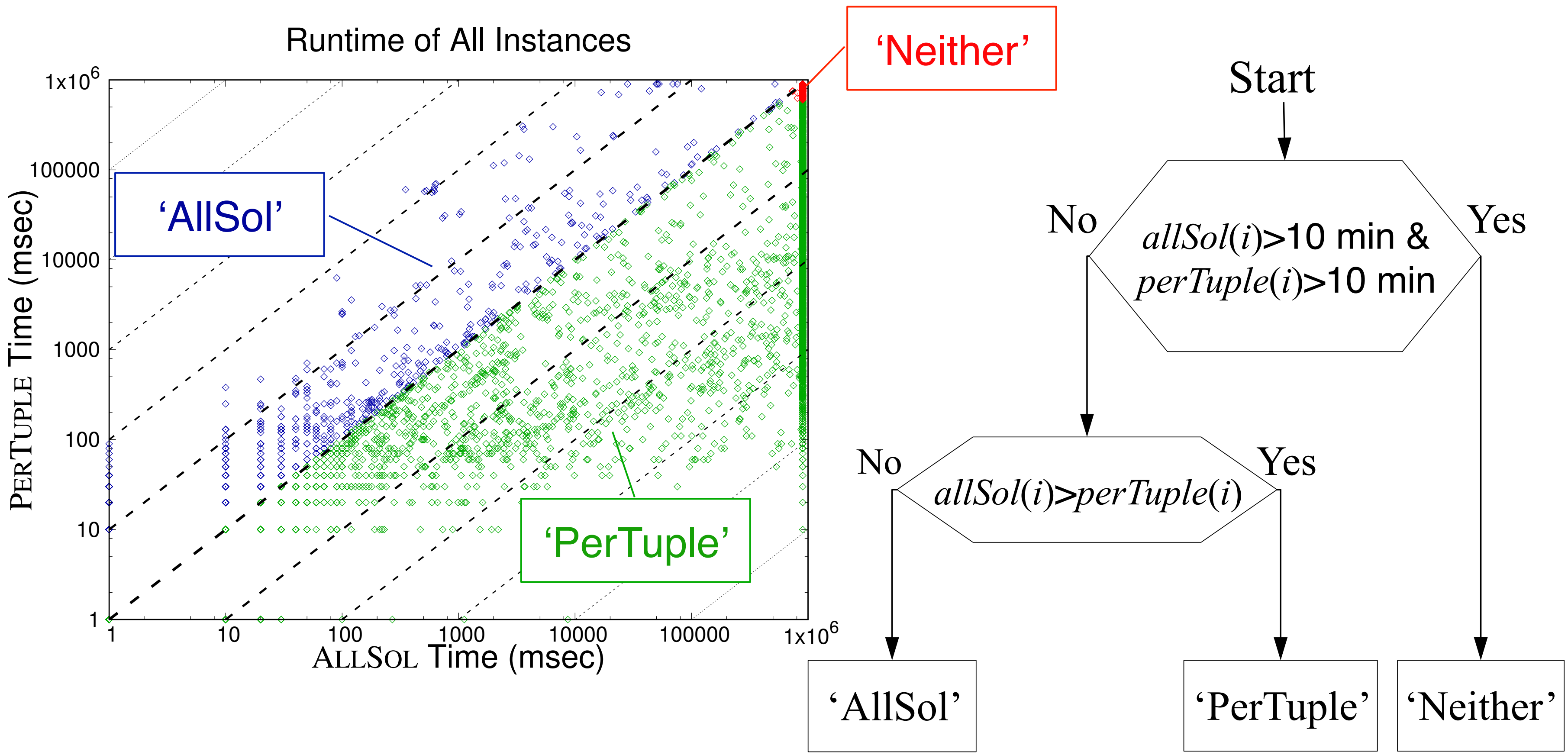
- Limits the time for processing a single cluster
- Allows recovery from a poor classification
- When interrupted, partial results of
  - PerTuple yield useful filtering
  - AllSol are useless



Classifier Training: Data

- 9362 individual clusters taken from 175 benchmarks
- For each cluster instance i, collected
  - The values of 73 classification features
  - The runtime of ALLSOL: $allSol(i)$
  - The runtime of PERTUPLE: $perTuple(i)$

Classifier Training: Labels



Classifier Training: Weights

- Weight of a training instance i, $weight(i)$

$$weight(i) = \begin{cases} w(allSol(i), perTuple(i)) & label(i) = \text{'}AllSol\text{'} \| \text{'}PerTuple\text{'} \\ 20 & label(i) = \text{'}Neither\text{'} \end{cases}$$

$$w(a,p) = \left\lceil \left| \log_{10}\left(\frac{a}{p}\right) \right| \cdot \left| \log_{10}\left(|a-p| + 0.01\right) \right| \right\rceil$$

- Designed to emphasize instance with both a
  - large proportional difference $\frac{a}{p}$
  - large absolute difference $|a-p|$

Classifier Training: Features

- CSP parameters
  - #variables, #constraints, #values, #tuples
  - Constraint arity, constraint tightness
  - Relational linkage
- Graph parameters: on dual, primal, and incidence graph
  - Density
  - Degree
  - Eccentricity
  - Clustering coefficient
- Using several descriptive statistics
  - min, max, mean, coefficient of variation, entropy

Classifier Training: Decision tree

- We built a decision tree classifier using the J48 algorithm from the Weka machine learning software
- Decision tree selected for:
  - Simplicity
  - Fast evaluation time
  - Only requires collection a subset of the features

Experiments: Set up

- Used 1055 instances from 42 benchmarks
- Backtrack search, dynamic *dom/deg* ordering
- Intel Xeon E5-2650 v3 2.30GHz processors with 12 GB memory
- 2 hours total time out per instance
- Compared GAC and six strategies (variants of FILTERCLUSTERS)

Experiments: Tested strategies

Algorithm	*classifier*	*interleaveGAC*	*timeout*
AllSol	Always select 'AllSol'	*false*	∞
PerTuple	Always select 'PerTuple'	*false*	∞
AllSol^{+}	Always select 'AllSol'	*true*	1 (s)
PerTuple^{+}	Always select 'PerTuple'	*true*	1 (s)
Random	Randomly select 'AllSol', 'PerTuple', or 'Neither'	*true*	1 (s)
DecTree	Decision tree selects 'AllSol', 'PerTuple', or 'Neither'	*true*	1 (s)

Experiments: Results

	GAC	AllSol	PerTuple	AllSol+	PerTuple+	Random	DecTree
Instances Completed	550	472	567	514	633	643	**685**
Average Time (s)	2,471	3,075	2,081	2,789	1,622	1,427	**1,121**

Conclusions

- A cluster-level portfolio, during lookahead
  - Is not only feasible, but also highly competitive
- Enforcing a timeout on consistency algorithms
  - Prevents getting stuck on one part of the problem
  - Does not affect soundness
- Future work
  - Dynamically determine timeout based on the anticipated amount of filtering
  - Heuristics for ordering the clusters

Thank you

Questions?

Algorithm 1: FilterClusters(*clusterOrder*, *classifier*, *interleaveGAC*, *timeout*)

```
Input: clusterOrder,classifier, interleaveGAC, timeout
Output: Entire problem is GAC with potentially minimal clusters
didFiltering ← true
passDidFiltering ← true
consistent ← true
(consistent, didFiltering) ←GAC()
if consistent = false then return false
while passDidFiltering do
    passDidFiltering ← false
    foreach cluster ∈ clusterOrder do
        algo ←Classify(cluster, classifier)
        if algo =‘AllSol’ then
            (consistent, didFiltering) ←AllSol(cluster, timeout)
        else if algo =‘PerTuple’ then
            (consistent, didFiltering) ←PerTuple(cluster, timeout)
        else didFiltering ← false
        if consistent = false then return false
        if didFiltering then passDidFiltering ← true
        if interleaveGAC and didFiltering then
            (consistent, didFiltering) ←GAC()
            if consistent = false then return false
    clusterOrder ←Reverse(clusterOrder)
if  interleaveGAC = false then
    (consistent, didFiltering) ←GAC()
    if consistent = false then return false
return true
```

		GAC		AllSol		PerTuple		AllSol+		PerTuple+		Random		DecTree				
Data Summary																		
#Completed 770/1055			550		472		567		514		633		643		**685**			
Average CPU time			2,471.6		3,075.3		2,081.9		2,789.4		1,622.7		1,427.4		**1,121.3**			
Sum of CPU time			1,900,653.4		2,364,878.9		1,601,010.4		2,145,062.1		1,247,840.7		1,097,633.8		**862,259.9**			
Benchmark		Hybrid solvers are best														A%	P%	N%
aim-100	21/24	17	>1,857.1	11	>3,984.0	20	>631.0	11	>3,741.4	21	559.3	16	>1,754.1	21	**512.4**	0	99	1
aim-200	17/24	8	>3,942.3	2	>6,494.0	8	>3,815.9	6	>5,208.7	10	>3,166.8	3	>5,990.8	14	**>1,647.3**	0	92	8
cmpsd-25-1-25	10/10	0	>7,200.0	9	>720.1	10	11.7	10	53.7	10	18.4	10	**0.1**	10	**0.1**	0	100	0
cmpsd-25-1-40	10/10	0	>7,200.0	8	>1,440.1	10	36.4	10	121.6	10	50.0	10	**0.1**	10	**0.1**	0	100	0
cmpsd-25-1-80	10/10	4	>4,445.6	6	>3,046.1	10	24.2	10	205.2	10	33.8	10	**2.2**	10	6.7	0	100	0
cmpsd-25-10-20	10/10	6	>2,892.7	0	>7,200.0	9	>2,208.1	0	>7,200.0	10	2,821.4	10	**404.8**	9	>2,202.6	0	96	4
cmpsd-75-1-25	10/10	0	>7,200.0	8	>1,440.4	8	>1,440.5	10	212.8	10	217.5	10	**3.1**	10	11.9	0	92	8
cmpsd-75-1-40	10/10	0	>7,200.0	6	>2,880.4	6	>2,880.4	10	611.5	10	454.0	10	**5.6**	10	64.4	0	93	7
cmpsd-75-1-80	10/10	3	>5,040.0	1	>6,480.1	2	>5,761.1	9	>2,241.4	10	1,176.6	10	**15.9**	10	123.5	0	99	1
cril	6/8	3	>3,968.4	3	>3,605.2	3	⋆>3,604.8	3	>3,606.0	4	⋆**>2,459.2**	4	⋆>2,999.9	3	⋆>3,604.9	2	66	32
ehi-90	100/100	84	>2,372.2	43	>4,456.8	72	>2,103.5	28	>5,259.4	81	>1,484.3	100	**61.2**	100	136.5	0	98	2
GC-hos	10/14	6	>2,882.3	0	⋆>7,200.0	3	⋆>5,129.9	2	⋆>6,360.4	7	⋆>3,401.4	8	⋆>2,693.5	8	⋆**>2,309.2**	0	98	2
GC-full-ins	24/41	17	>2,105.7	4	⋆>6,004.0	17	⋆>2,440.0	8	⋆>5,266.7	18	⋆>2,146.5	15	⋆>3,008.7	22	⋆**>1,010.9**	0	99	1
GC-mug	8/8	4	>3,600.0	6	>2,182.2	6	>2,156.0	8	47.8	8	**41.5**	4	>3,600.0	8	102.6	0	97	3
pseudo-aim	42/48	25	>2,917.5	20	>3,867.3	28	>2,406.8	24	>3,676.8	37	>1,054.0	28	>2,515.5	42	**265.4**	0	94	6
QCP-15	15/15	10	>3,023.7	2	>6,241.1	2	>6,241.4	2	>6,250.4	3	>6,041.4	8	>3,973.8	15	**533.3**	0	80	20
rand-8-20-5	20/20	19	>1,532.7	3	>6,551.8	0	>7,200.0	18	>2,333.3	3	>6,811.0	20	**587.8**	20	**605.2**	35	59	7
rlfapGraphsMod	11/12	5	>3,975.5	4	>4,582.2	5	>4,180.5	7	>4,015.7	9	>1,878.8	11	**843.3**	8	>2,043.0	0	88	12
rlfapScens11	7/12	0	>7,200.0	3	>4,199.1	4	>3,373.2	5	>3,528.1	7	**1,016.4**	6	>1,371.2	1	>6,183.0	15	60	25
rlfapScensMod	13/13	7	>3,323.4	8	>3,103.4	9	>2,316.1	8	>3,227.5	10	>2,008.9	12	**>1,249.1**	10	>2,227.1	7	81	12
No clear winner																		
aim-50	24/24	24	**0.6**	24	6.2	24	2.3	24	53.9	24	**0.7**	24	4.7	24	**0.6**	0	100	0
cmpsd-25-1-2	10/10	0	>7,200.0	10	**0.1**	10	**0.1**	10	**0.1**	10	**0.1**	10	**0.1**	9	>720.1	0	94	6
cmpsd-75-1-2	10/10	0	>7,200.0	10	**0.5**	10	**0.6**	10	**0.5**	10	**0.6**	10	**0.6**	10	**0.6**	0	86	14
hanoi	5/5	5	**1.8**	5	**2.5**	5	**2.5**	5	**2.5**	5	**2.5**	5	**2.5**	5	**2.5**	0	100	0
knights	11/19	10	**>1,098.3**	7	>2,716.0	8	>2,485.4	10	**>1,144.4**	10	**>1,138.0**	10	**>1,128.6**	10	**>1,131.3**	0	64	36
modRenault	50/50	27	>3,439.2	50	**2.1**	50	**3.1**	50	**2.4**	50	3.2	50	**2.6**	48	>290.7	12	84	3
rand-10-20-10	20/20	20	3.7	20	**1.0**	20	**1.1**	20	**1.0**	20	**1.1**	20	**1.1**	20	**1.3**	0	100	0
ssa	7/8	6	**>1,029.3**	6	**>1,058.6**	6	**>1,058.8**	6	**>1,052.3**	6	**>1,052.5**	5	>2,058.1	6	**>1,065.3**	0	96	4
Basic solvers are best																		
dag-rand	25/25	25	2,467.6	25	**21.0**	25	**21.7**	25	45.2	25	38.2	25	24.7	24	>1,423.5	3	96	1
dubois	7/13	7	**1,959.6**	6	>2,191.7	7	2,099.1	6	>2,175.1	6	>2,085.8	5	>3,388.8	6	>2,457.6	0	100	0
GC-reg-fpsol	8/37	6	**>1,814.4**	4	⋆>4,237.4	4	⋆>4,238.9	2	⋆>5,407.9	2	⋆>5,408.1	2	⋆>5,408.1	2	⋆>5,408.2	0	99	1
GC-reg-inithx	7/32	5	**>2,129.3**	4	⋆>3,915.8	2	⋆>5,160.2	2	⋆>5,159.6	2	⋆>5,160.1	2	⋆>5,160.2	2	⋆>5,160.1	0	100	0
GC-reg-mulsol	13/49	9	**>2,218.0**	9	>2,928.4	9	⋆>2,928.9	5	>4,440.2	5	⋆>4,440.5	5	⋆>4,440.5	5	⋆>4,440.5	0	99	1
GC-reg-zeroin	8/31	6	**>1,801.6**	5	>3,249.9	5	⋆>3,251.2	3	>4,519.0	3	⋆>4,519.5	3	⋆>4,519.5	3	⋆>4,519.6	0	90	10
GC-sgb-book	23/26	18	>1,818.3	19	>1,657.5	23	**256.2**	16	>3,106.7	22	>737.5	20	>1,321.6	22	>657.4	0	94	6
GC-sgb-games	4/4	2	>3,600.2	2	>3,600.2	4	**26.0**	2	>3,600.2	4	46.4	3	>1,999.1	4	46.4	0	99	1
GC-sgb-miles	13/42	11	**>1,411.7**	9	⋆>2,749.8	8	⋆>3,109.3	6	⋆>3,902.3	6	⋆>3,883.6	7	⋆>3,488.3	7	⋆>3,652.3	0	87	13
GC-sgb-queen	14/50	10	**>2,619.2**	6	>4,676.4	7	⋆>3,874.8	3	>5,852.4	6	⋆>4,421.7	6	⋆>4,315.4	9	⋆>3,415.8	0	76	24
haystacks	8/51	5	>2,786.7	7	>1,055.8	8	**228.6**	5	>2,700.9	7	>1,043.3	5	>2,716.0	7	>934.3	0	100	0
marc	10/10	10	**16.8**	10	253.6	0	⋆>7,200.0	10	1,321.7	0	⋆>7,200.0	0	⋆>7,200.0	0	⋆>7,200.0	-	-	-
os-taillard-4	29/30	27	**>887.8**	2	>6,704.7	2	>6,704.7	21	>2,427.0	24	>2,967.6	23	>1,876.7	23	>2,681.4	15	83	1
tightness0.9	99/100	99	**352.6**	85	>1,946.3	98	>489.7	84	>1,950.5	98	>561.8	98	>741.6	98	>549.5	0	99	0

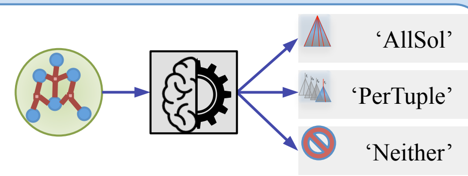
Classifier Training: Evaluation

- Using 10-fold cross validation
- Using both weighted and un-weighted instances

	weighted	**unweighted**
Accuracy	90.8%	80.1%
F-Measure		
'AllSol'	0.50	0.40
'PerTuple'	0.89	0.85
'Neither'	0.93	0.93

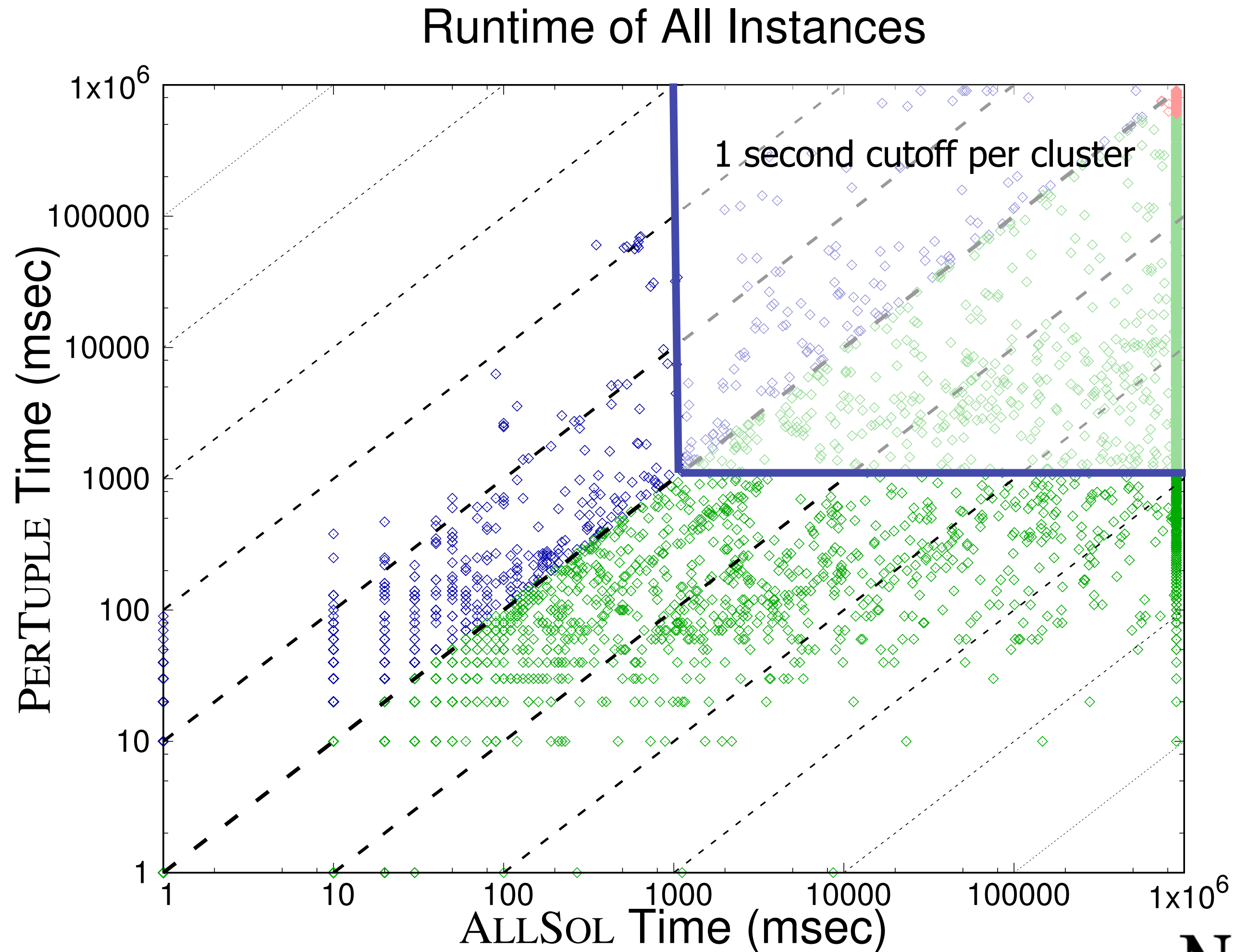
FILTERCLUSTERS

- Enforce GAC globally
- Build cluster ***LIST***
- Repeat until quiescence
  - For cluster ***C*** in ***LIST***
    - Classify ***C***

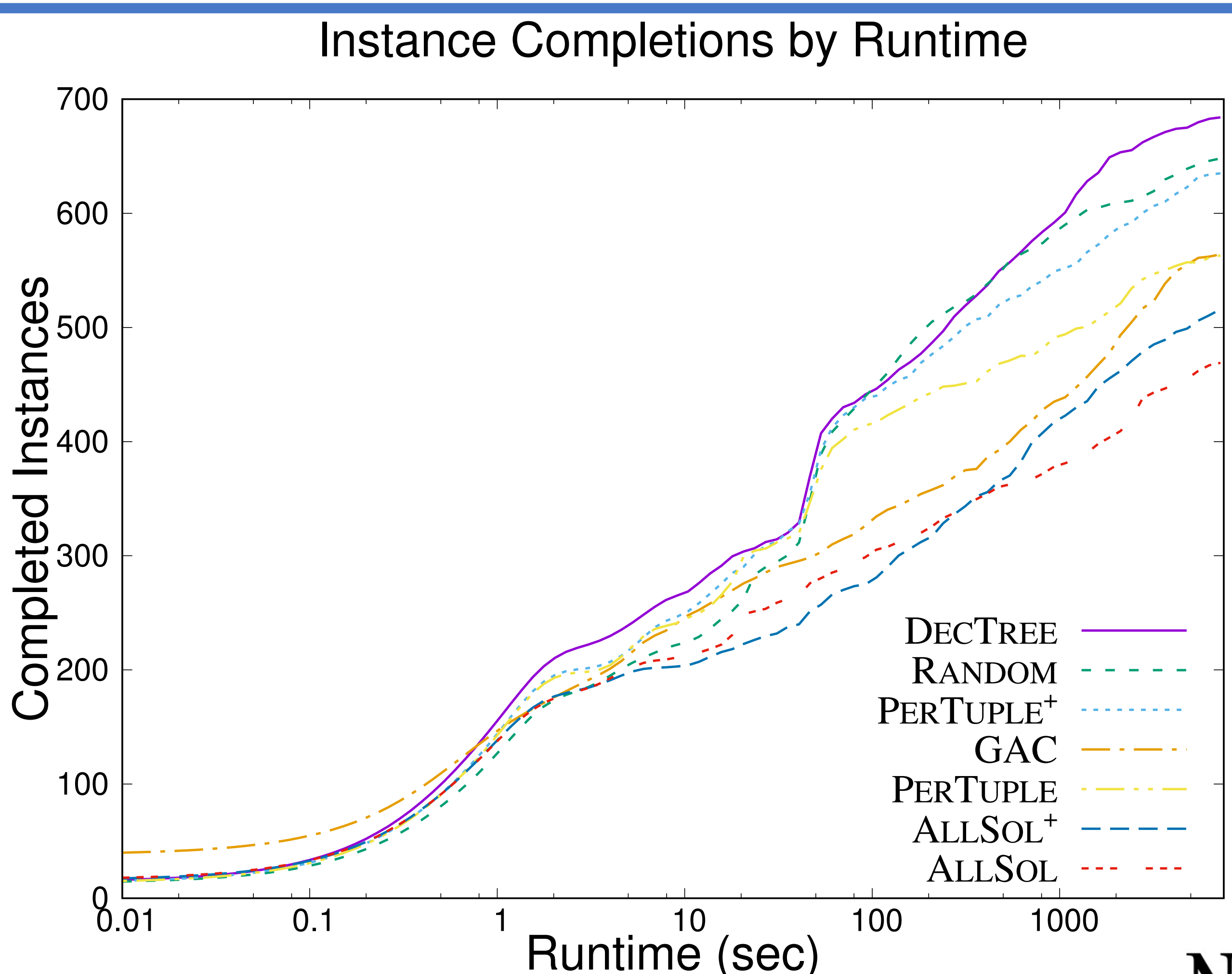


    - Process ***C*** within time limit
    - Enforce GAC globally
  - Reverse ***LIST***

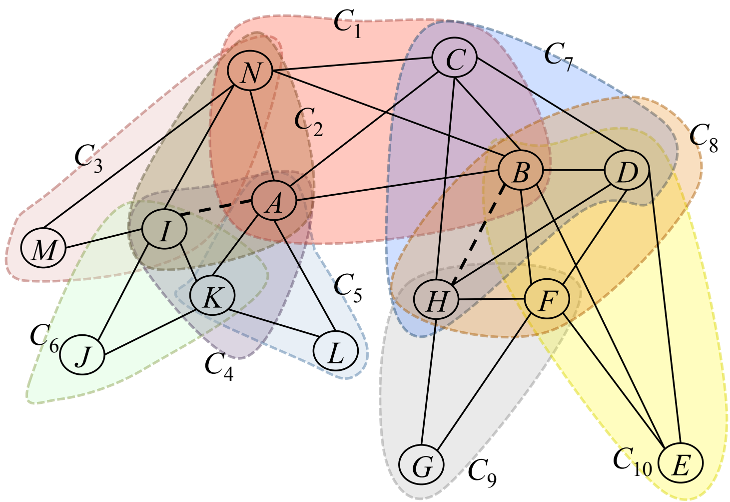
Experiments: Tested strategies (2)



Experiments: Results (2)



Background: Tree decomposition, minimality



- Build a tree decomposition
- Localize the enforcement of minimality to the clusters
- Process clusters in MAXCLIQUES order back and forth to quiescence