Exploiting Global Properties in Path-Consistency Applied on SAT

Pavel Surynek

Department of Theoretical Computer Science and Mathematical Logic

Faculty of Mathematics and Physics
Charles University in Prague

http://ktiml.mff.cuni.cz/~surynek

Constraint Satisfaction Problem (CSP)

- Constraint satisfaction problem over the universe of elements $\mathbb{D}$ is a triple **(X,C,D)**
  - **X** – finite set of variables
  - **C** – finite set of constraints
  - **D** – is a function $D:X \rightarrow \mathcal{P}(\mathbb{D})$
  - each constraint $c \in C$ is a construct of the form $<(x_1^c, x_2^c, \ldots, x_{k(c)}^c), R^c>$
    - k(c) is arity of the constraint
    - $x_i^c \in X$ for I = 1,2,…,k(c) and $R^c \subseteq D(x_1^c) \times D(x_2^c) \times \ldots \times D(x_{k(c)}^c)$

example: $\mathbb{D}=\{1,2,3\}$
$X=\{a,b,c\}$
$C=\{<(a,b),\text{"<"}>;$
$<(b,c),\text{"="}>\}$
$D(a)=D(b)=D(c)=\mathbb{D}$

- The task is to find **assignment of values to variables** from their domains such that all the constraints are satisfied
  - or decide that **no** such **valuation exists**

example: a=1, b=2, c=2

- Decision variant is an **NP-complete** problem

Boolean Satisfiability (SAT)

- A **Boolean formula** is given - variables can take either the value ***TRUE*** or ***FALSE***

example: $(\neg x \Rightarrow \neg y) \wedge (x \Rightarrow \neg y)$

- The task is to find **valuation of variables** such that the formula is **satisfied**
  - or decide that **no** such **valuation exists**

example: x = *TRUE*
y = *FALSE*

- Conjunctive normal form (**CNF**) - standard form of the input formula for SAT solvers
  - **variables**: $x_1, x_2, x_3, \ldots$
  - **literals**: $x_1, \neg x_1, x_2, \neg x_2, \ldots$ variable or its negation
  - **clauses**: $(x_1 \vee \neg x_2 \vee \neg x_3)$ ... disjunction of literals
  - **formula**: $(x_1 \vee \neg x_2) \wedge (x_1 \vee x_2 \vee \neg x_3)$ ... conjunction of clauses

example:
p cnf 3 2
1 -2 0
1 2 -3 0
...

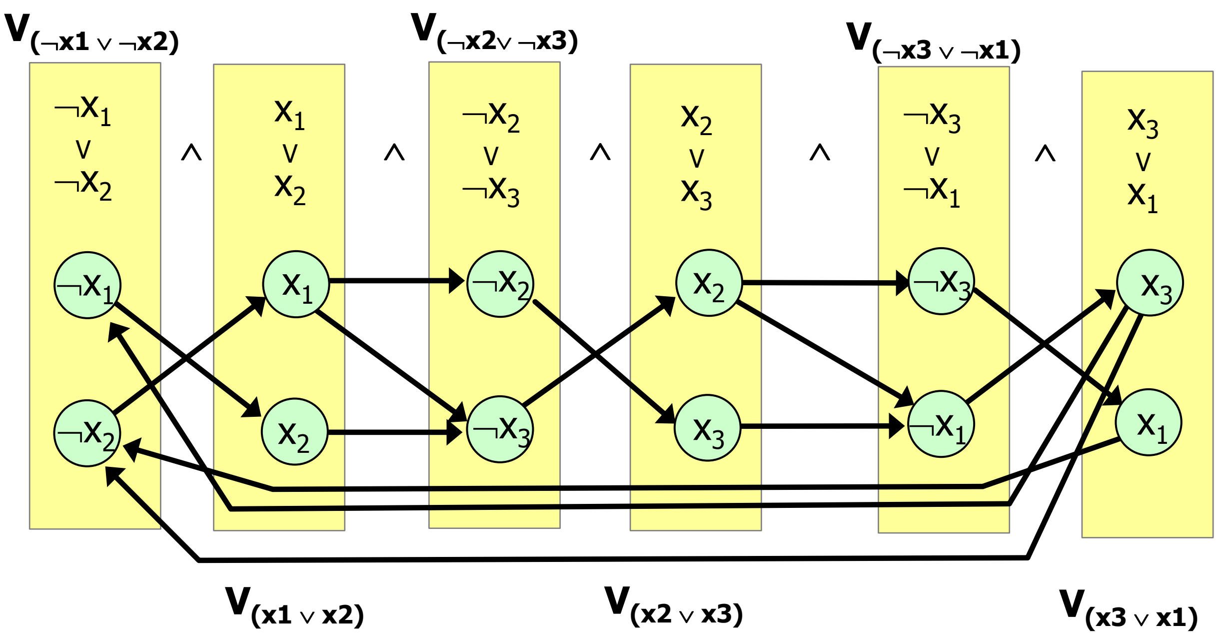
- Clauses represent constraints that must be all satisfied (can be regarded as CSP) – SAT and CSP are mutually reducible

Motivation for Global Consistencies

- CSP paradigm provides many types of **local consistencies**
  - local inference is typically **too weak** for SAT
  - arc-consistency, path-consistency, i,j-consistency
    - insignificant gain in comparison with unit-propagation
    - expensive propagation with respect to the inference strength
- **Global** consistencies (global constraints)
  - provide strong global inference
    - often leads to significant simplification of the problem
  - application of **global consistencies** in SAT is quite rare
- Consistency based on **structural properties**
  - interpret SAT as a graph and find graph structures

Path-consistency in Literal Encoding (1)

- SAT as CSP: **Literal encoding** model (X,C,D)
  - X ... variables ↔ clauses, C ... constraints ↔ values standing for complementary literals are forbidden, D ... variable domains ↔ literals
- Interpret path-consistency in the CSP model of SAT as a **directed graph**
  - **vertices** ↔ values in domains, **edges** ↔ allowed pairs of values



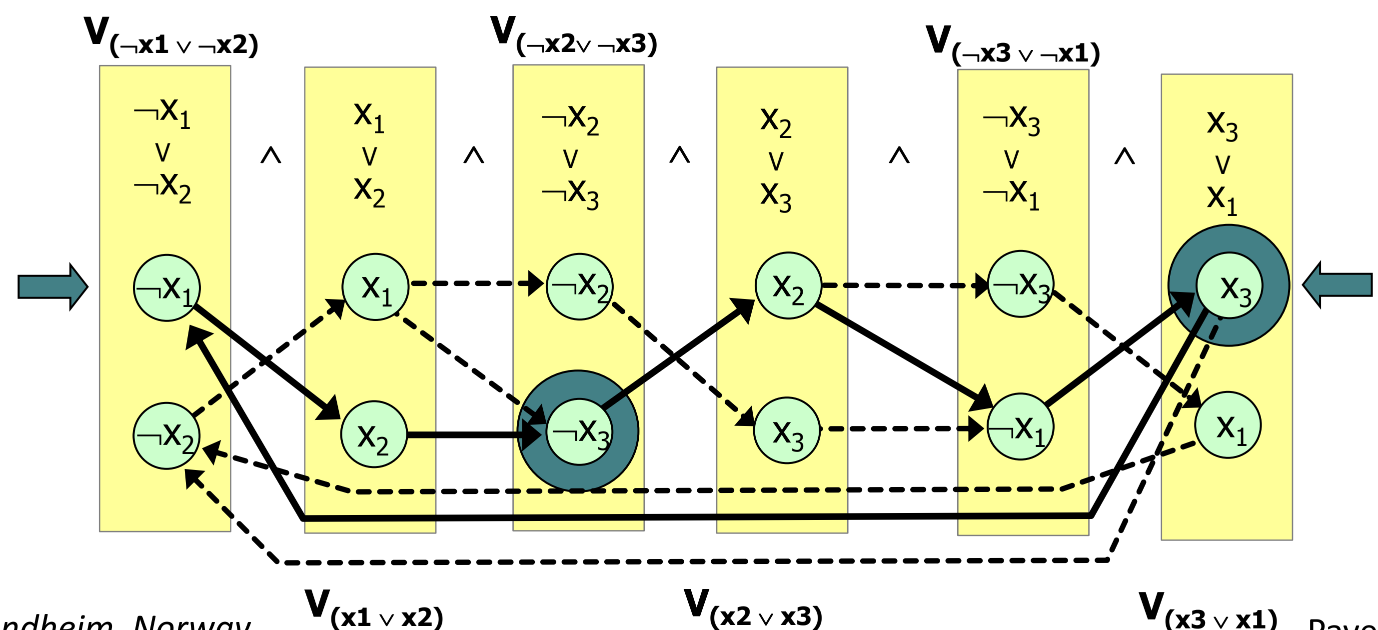
example:
$X=V_{(\neg x1 \vee \neg x2)}, V_{(x1 \vee x2)}, \ldots$

example:
$D(V_{(\neg x1 \vee \neg x2)})=\{\neg x_1, \neg x_2\}$

example:
$V_{(\neg x1 \vee \neg x2)} = \neg x_1$ and
$V_{(x1 \vee x2)} = x_1$
is forbidden

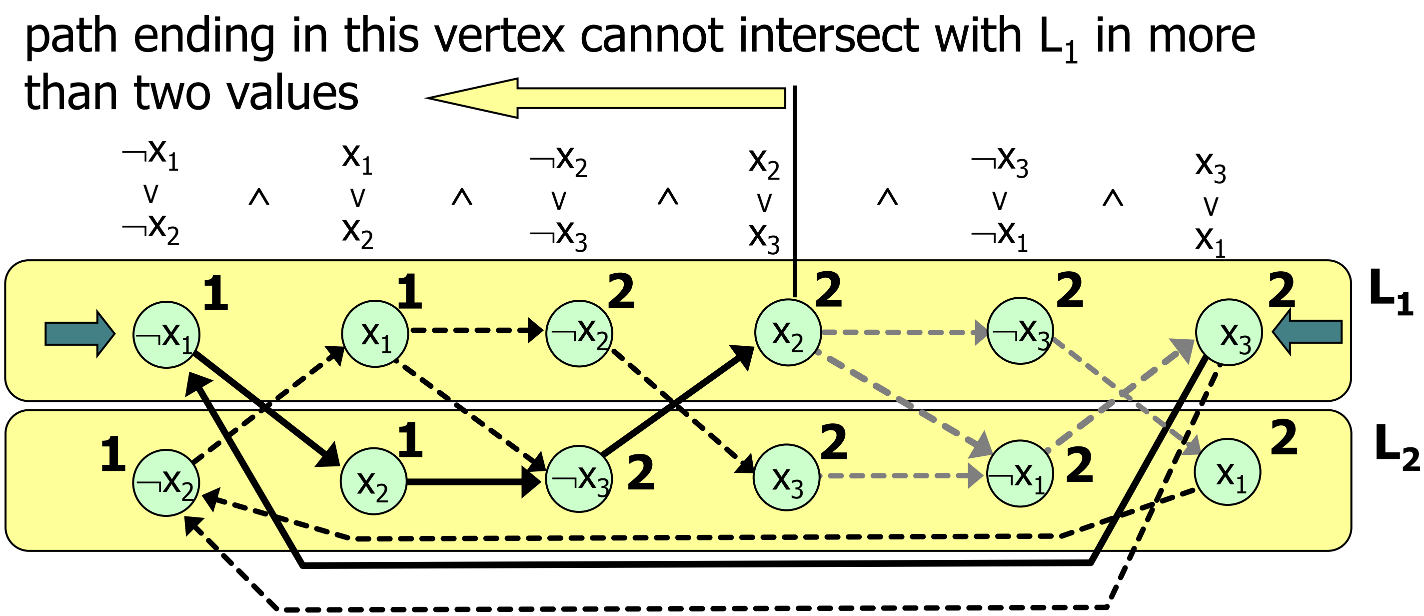
Path-consistency in Literal Encoding (2)

- Let us have a **sequence of variables (path)**
  - a pair of values is **path-consistent** w.r.t. to the sequence if there is an oriented path connecting them in the graph interpretation going through the sequence and values itself are connected
- **Ignores** constraints between non-neighboring variables in the sequence of variables



Modified Path-Consistency for SAT

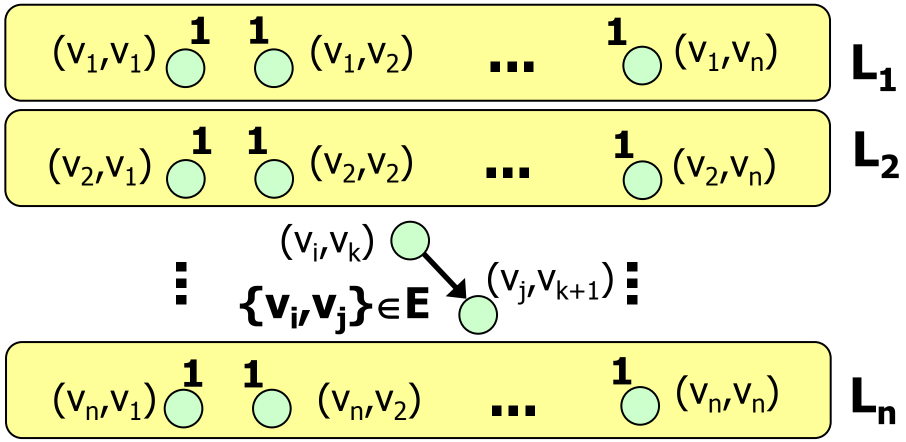
- Deduce **more information from constraints**
  - decompose values into **disjoint sets** (called **layers** ... L_1, L_2,..., L_M)
  - **deduce more information** from constraints - calculate **maximum size of the intersection of the constructed path with individual layers –** denoted as χ
- Stronger restriction on paths ► **stronger propagation**



NP-completeness of the Modified Path Consistency

- Enforcing **modified path-consistency** is **difficult** (NP-complete)
  - The decision problem is whether there exists a path conforming to the maximum size of the intersection with individual layers.
- **Lemma:** The decision variant of the problem belongs to the NP class.
  - The path is of polynomial size with respect to the graph interpretation.
  - It can be checked in polynomial time whether the path conforms to the maximum size of intersection with individual layers.
- **Lemma:** The existence of a **Hamiltonian path** in a graph is reducible to the existence of a path conforming to the maximum size of intersection with layers.

- **Main idea** of the proof: **G=(V,E)**, where $V=\{v_1,v_2,\ldots,v_n\}$
  (i) Construct an instance of modified path consistency in the form of a matrix
- **(ii)** Associate rows of the matrix with layers and set the maximum size of the intersection to 1



Intersection Matrices

- An **intersection matrix** is defined for each value in the graph interpretation of path-consistency – it is denoted as ψ(v)
  - Let L_1, L_2, …, L_M be a layer decomposition of the graph interpretation
  - Let K be the number of variables involved in the path
  - ▶ The **intersection matrix** is of type M × (K+1)
- Intersection matrix **ψ(v)** w.r.t. a pair of values v_0 and v_K
  - $\boldsymbol{\psi(v)_{i,j}}$ represents the number of paths starting in v_0 and ending in v that **partially** conforms to maximum sizes of intersection with layers such that they intersect with L_i j-times.
- It is not possible to enforce exact conformity to calculated maximum sizes of intersection with layers
  - Therefore we need to talk about partial conformity.

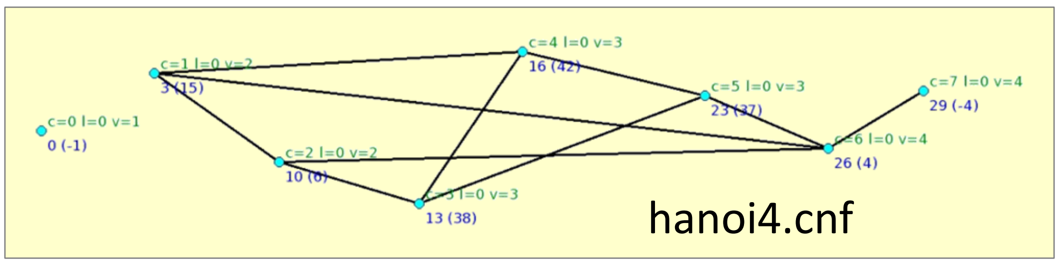
Intersection Matrix Calculation

- **Intersection matrix** can be updated easily
  - $\psi(v)$ is calculated from $\psi(u_1)$, $\psi(u_2)$,..., $\psi(u_m)$ where u_1, u_2,..., u_m are a values from the domain of the **previous variable** in the path
- If it is detected that **no** of the paths starting in $\mathbf{v_0}$ and ending in **v** conforms to the maximum size of the intersection with the layer L_i such that $v \in L_i$ then $\psi(v)$ is set to 0 (matrix)
  - maximum intersection sizes with other layers cannot be violated since intersection size with them does no change
  - **relaxation:** paths that do not conform to maximum sizes of intersections with layers are propagated further

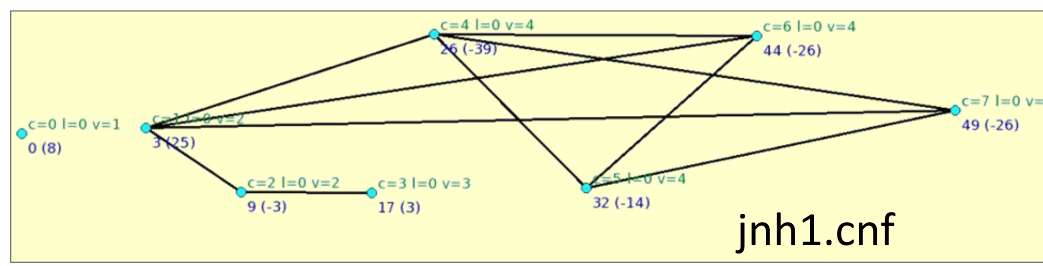
Visualization of Layers

using GraphExplorer software (Surynek, 2007-2010)

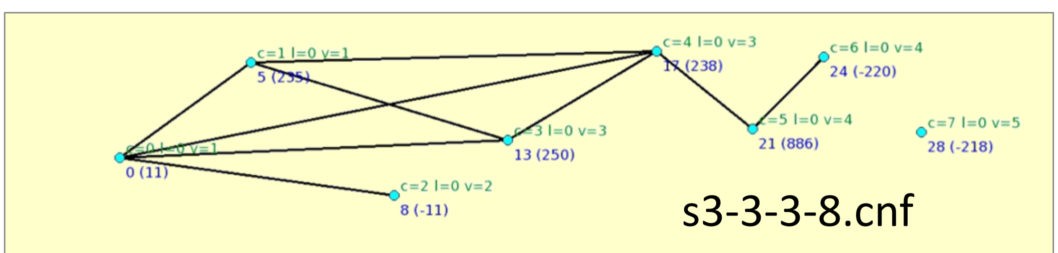
- Layer decomposition was constructed with several **most constrained clauses**
  - several benchmark problems from the **SAT Library**



hanoi4.cnf



jnh1.cnf



s3-3-3-8.cnf

Experimental Evaluation (1)

- Modified path-consistency on **pigeon-hole** instances (decisions)
  - standard path-consistency is unable to infer any new clause

SAT instance	Instance characteristics		Inferred binary clauses		Minisat2 decisions		
	Variables	Clauses	PC	mPC	Original	PC	mPC
hole6	42	133	0	42	1777	1777	1
hole7	56	204	0	56	10123	10123	1
hole8	72	297	0	72	40554	40554	1
hole9	90	415	0	90	202160	202160	1
chnl10_11	220	1122	0	220	N/A	N/A	1
chnl10_12	240	1344	0	240	N/A	N/A	1
chnl10_13	260	1586	0	260	N/A	N/A	1
chnl11_12	264	1476	0	264	N/A	N/A	1
chnl11_13	286	1742	0	286	N/A	N/A	1
chnl11_20	440	4220	0	440	N/A	N/A	1
fpga10_12_uns_rcr	240	1344	0	240	N/A	N/A	1
fpga10_13_uns_rcr	260	1586	0	260	N/A	N/A	1
fpga10_15_uns_rcr	300	2130	0	300	N/A	N/A	1
fpga10_20_uns_rcr	400	3840	0	400	N/A	N/A	1
fpga11_11_uns_rcr	264	1476	0	264	N/A	N/A	1
fpga11_12_uns_rcr	286	1742	0	286	N/A	N/A	1

Experimental Evaluation (2)

- Modified path-consistency on **pigeon-hole** instances (runtime)
  - binary clauses inferred by modified path-consistency can help the SAT solver to decide the instance **immediately**

SAT instance	Preprocessing runtime (sec.)		Minisat2 solving runtime (sec.)			Total solving runtime (sec.)	
	Runtime PC (sec.)	Runtime mPC (sec.)	Original	PC	mPC	PC	mPC
hole6	0.01	0.04	0.00	0.00	0.00	0.01	0.04
hole7	0.02	0.14	0.10	0.10	0.00	0.12	0.14
hole8	0.04	0.32	0.48	0.48	0.00	0.52	**0.32**
hole9	0.07	0.64	3.61	3.61	0.00	3.68	**0.64**
chnl10_11	0.23	2.38	> 10.0	> 10.0	0.00	> 10.0	**2.38**
chnl10_12	0.25	2.6	> 10.0	> 10.0	0.00	> 10.0	**2.6**
chnl10_13	0.27	2.82	> 10.0	> 10.0	0.00	> 10.0	**2.82**
chnl11_12	0.36	4.18	> 10.0	> 10.0	0.00	> 10.0	**4.18**
chnl11_13	0.39	4.54	> 10.0	> 10.0	0.00	> 10.0	**4.54**
chnl11_20	0.63	7.05	> 10.0	> 10.0	0.00	> 10.0	**7.05**
fpga10_12_uns_rcr	0.25	2.61	> 10.0	> 10.0	0.00	> 10.0	**2.61**
fpga10_13_uns_rcr	0.28	2.82	> 10.0	> 10.0	0.00	> 10.0	**2.82**
fpga10_15_uns_rcr	0.32	3.27	> 10.0	> 10.0	0.00	> 10.0	**3.27**
fpga10_20_uns_rcr	0.45	4.37	> 10.0	> 10.0	0.00	> 10.0	**4.37**
fpga11_11_uns_rcr	0.36	4.18	> 10.0	> 10.0	0.00	> 10.0	**4.18**
fpga11_12_uns_rcr	0.39	4.54	> 10.0	> 10.0	0.00	> 10.0	**4.54**

Experimental Evaluation (3)

- (2,k)-consistency on **integer factorization** (decisions)
  - can reduce the number of decisions significantly

SAT instance	Instance characteristics		Inferred binary clauses		Minisat2 decisions		
	Variables	Clauses	PC	(2,k)-c	Original	PC	(2,k)-c
difp_19_0_arr_rcr	1201	6563	0	675	142710	142710	61317
difp_19_0_wal_rcr	1755	10446	103	281	73018	339662	25340
difp_19_1_arr_rcr	1201	6563	6	307	250692	87894	81144
difp_19_1_wal_rcr	1755	10446	363	561	129235	133055	77039
difp_19_2_wal_rcr	1755	10446	38	212	288500	207775	98374
difp_19_3_arr_rcr	1201	6563	128	342	114648	122379	100824
difp_19_3_wal_rcr	1755	10446	36	202	609247	968223	109741
difp_20_0_arr_rcr	1201	6563	91	754	8174	39097	12598
difp_20_0_wal_rcr	1755	10446	378	553	65601	752497	123562
difp_20_1_wal_rcr	1755	10446	10	131	362145	540378	147005
difp_20_2_arr_rcr	1201	6563	57	611	62119	438572	49700
difp_20_2_wal_rcr	1755	10446	866	2375	184778	177142	15415
difp_20_3_arr_rcr	1201	6563	0	73	142823	142823	89801
difp_20_3_wal_rcr	1755	10446	357	5798	26905	159962	45492

Experimental Evaluation (4)

- (2,k)-consistency on **integer factorization** (runtime)
  - runtime is can be reduced by preprocessing as well

SAT instance	**Preprocessing time** (sec.)		**Minisat2 solving time** (sec.)			**Total solving time** (sec.)	
	Runtime PC	Runtime (2,k)-c	Original	PC	(2,k)-c	PC	(2,k)-c
difp_19_0_arr_rcr	3.15	3.19	27.78	27.78	9.63	30.93	**12.82**
difp_19_0_wal_rcr	3.74	3.58	10.97	68.94	2.68	72.68	**6.26**
difp_19_1_arr_rcr	3.00	3.06	54.17	15.99	14.21	**18.99**	**17.27**
difp_19_1_wal_rcr	3.56	3.48	24.19	24.86	14.21	28.42	**17.69**
difp_19_2_wal_rcr	3.58	3.43	65.13	41.53	18.85	**45.11**	**22.28**
difp_19_3_arr_rcr	3.00	3.57	20.59	22.80	16.86	25.80	**20.43**
difp_19_3_wal_rcr	3.79	3.48	164.05	286.71	19.59	290.50	**23.07**
difp_20_0_arr_rcr	3.04	3.43	0.73	5.69	1.11	8.73	12.16
difp_20_0_wal_rcr	3.64	3.62	10.48	208.25	23.45	211.89	27.07
difp_20_1_wal_rcr	3.99	3.69	83.49	134.27	28.22	137.96	**31.91**
difp_20_2_arr_rcr	3.01	3.36	9.57	108.28	7.40	111.29	10.76
difp_20_2_wal_rcr	23.3	25.6	38.49	37.47	1.62	60.77	**27.22**
difp_20_3_arr_rcr	17.33	19.6	27.73	27.73	16.78	45.06	36.38
difp_20_3_wal_rcr	21.94	23.8	3.54	31.27	7.33	53.21	31.13

Conclusion and Future Work

- The **modification** of path-consistency in order to make it stronger by incorporating **global** knowledge
  - exploits **more global** inference than the standard version
  - **non-neighboring variables** in the path are taken into account
  - **maximum intersection** sizes with layers (subsets of values) is used as pruning condition
- Additional experimental evaluation shown a potential for future work
  - further increasing of inference strength towards (2,k)-consistency
  - keep low computational costs and preserve global reasoning

References

- Chmeiss, A., Jégou, P.: Efficient Constraint Propagation with Good Space Complexity. Proceedings of the Second International Conference on Principles and Practice of Constraint Programming (CP 1996), pp. 533-534, LNCS 1118, Springer, 1996.
- Cook, S. A.: The Complexity of Theorem Proving Procedures. Proceedings of the 3rd Annual ACM Symposium on Theory of Computing (STOC 1971), pp. 151-158, ACM Press, 1971.
- Dowling, W., Gallier, J.: Linear-time algorithms for testing the satisfiability of propositional Horn formulae. Journal of Logic Programming, Volume 1 (3), 267-284, Elsevier Science Publishers, 1984.
- Holger, H. H., Stützle, T.: SATLIB: An Online Resource for Research on SAT. Proceed-ings of Theory and Applications of Satisfiability Testing, 4th International Conference (SAT 2000), pp.283-292, IOS Press, 2000, http://www.satlib.org, [October 2010].
- Mohr, R., Henderson, T. C.: Arc and Path Consistency Revisited. Artificial Intelligence, Volume 28 (2), 225-233, Elsevier Science Publishers, 1986.
- Montanari, U.: Networks of constraints: Fundamental properties and applications to picture processing. Information Sciences, Volume 7, pp. 95-132, Elsevier, 1974.
- Surynek, P.: Making Path Consistency Stronger for SAT. Proceedings of the Annual ERCIM Workshop on Constraint Solving and Constraint Logic Programming (CSCLP 2008), ISTC-CNR, 2008.
- Surynek, P.: An Adaptation of Path Consistency for Boolean Satisfiability: a Theoretical View of the Concept. Proceedings of the Annual ERCIM Workshop on Constraint Solving and Constraint Logic Programming, 2010 (CSCLP 2010), pp. 16-30, Fraunhofer FIRST, 2010.
- Surynek, P.: Between Path-Consistency and Higher Order Consistencies in Boolean Satisfiability. Annual ERCIM Workshop on Constraint Solving and Constraint Logic Programming, 2011 (CSCLP 2011), York, United Kingdom, pp. 120-134, University of York, 2011.