Unate Covering, Binate Covering, Graph Coloring Maximum Cliques

part B

Combinational Problems: Unate Covering, Binate Covering, Graph Coloring and Maximum Cliques

Unit 6

part B

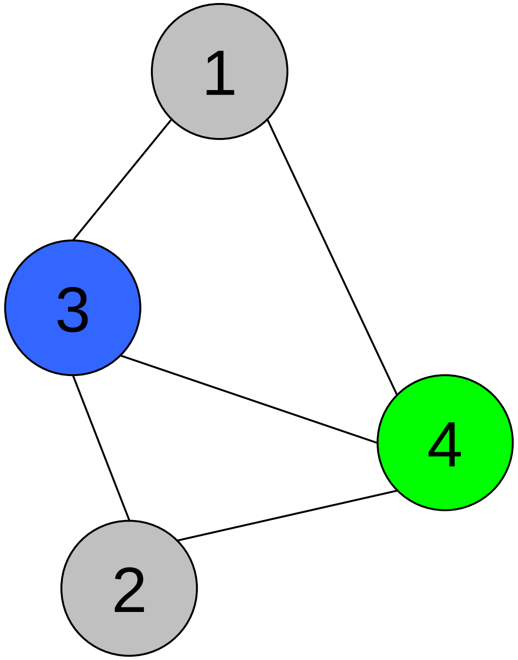
Column Multiplicity

Bound Set

AB \ CD	00	01	11	10
00	0	0	–	1
01	–	1	0	0
11	1	–	1	0
10	1	1	0	0
	1	2	3	4

Free Set

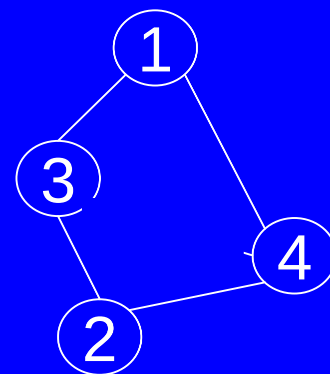
f



Column Multiplicity-other example

Bound Set (CD), Free Set (AB)

AB \ CD	00	01	11	10
00	0	0	–	1
01	–	1	0	0
11	1	–	1	-
10	1	1	0	0
	1	2	3	4



X

C \ D	0	1
0	0	0
1	1	1

X=G(C,D)

X=C in this case

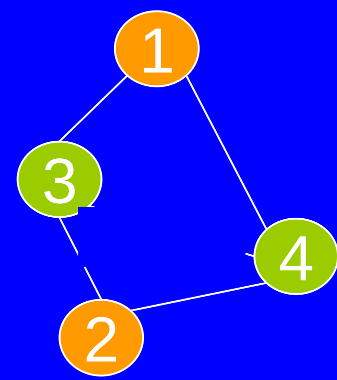
But how to calculate function H?

Column Multiplicity-other example

Bound Set

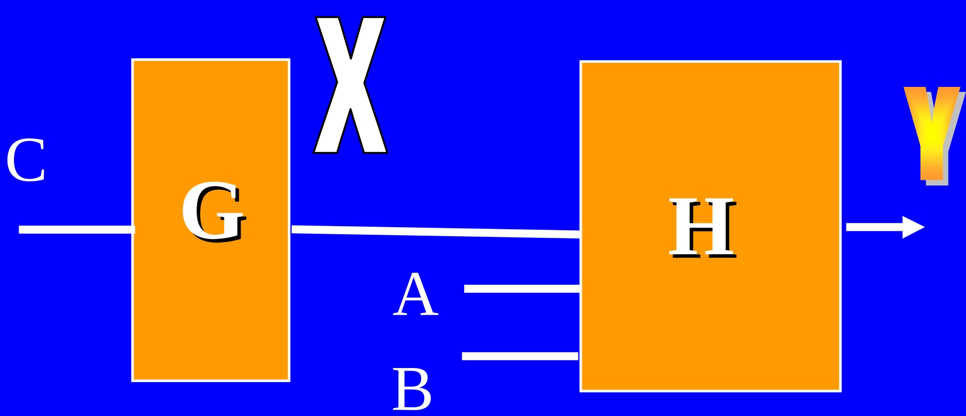
AB \ CD	00	01	11	10
00	0	0	–	1
01	–	1	0	0
11	1	–	1	-
10	1	1	0	0
	1	2	3	4

Free Set



C \ D	0	1
0	0	0
1	1	1

X



AB \ X	0	1
00	0	1
01	1	0
11	1	1
10	1	0

Y

Y=G(X,A,B)

Basic Definitions

Definition 1. Node A in the incompatibility graph ***covers*** node B if

1) A and B have no common edges;
2) A has edges with all the nodes that B has edges with;
3) A has at least one more edge than B.

Basic Definitions (cont'd)

Definition 2. If conditions 1) and 2) are true and A and B have the same number of nodes, then it is called ***pseudo-covering.***

Definition 3. The ***complete graph*** is one in which all the pairs of vertices are connected.

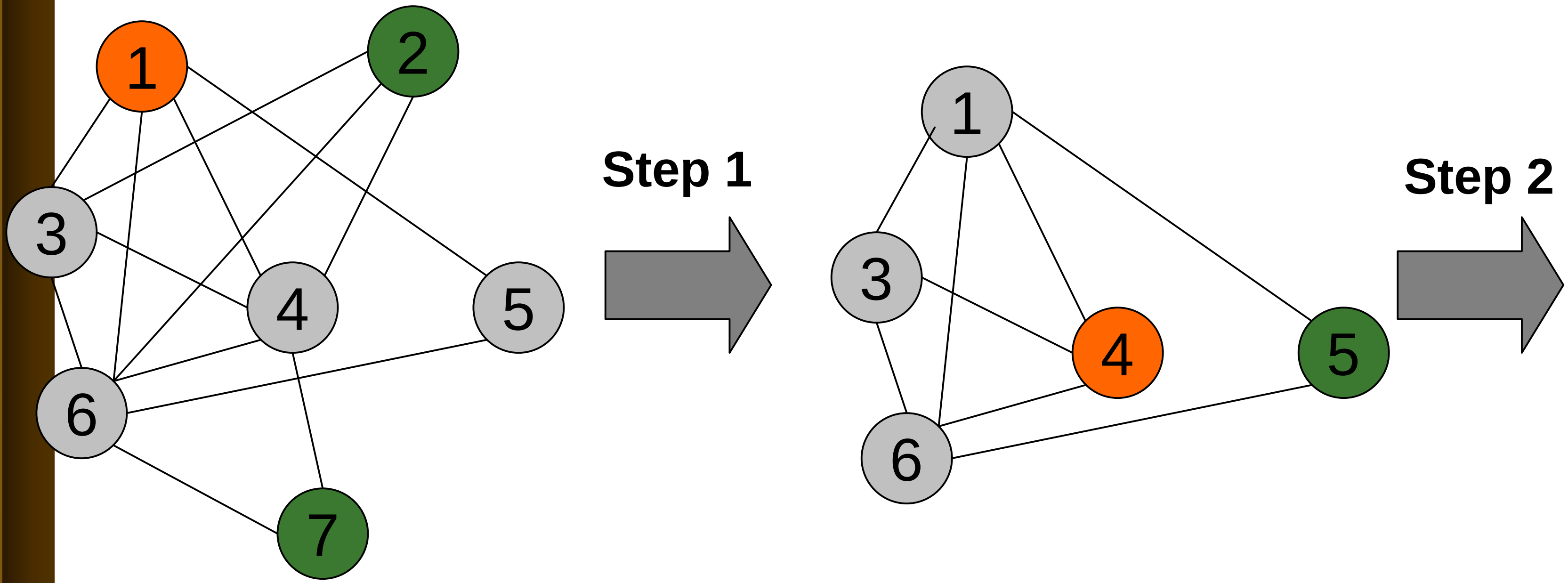
Definition 4. A ***non-reducible graph*** is a graph that is not complete and has no covered or pseudo-covered nodes. Otherwise, the graph is ***reducible***.

New Algorithm DOM for Graph Coloring by Domination Covering

Theorem 1. If any node A in the incompatibility graph covers any other node B in the graph, then ***node B can be removed from the graph***, and in a pseudo-covering any one of the nodes A and B can be removed.

Theorem 2. If a graph is reducible and can be reduced to a complete graph by successive removing of all its covered and pseudo-covered nodes, then ***Algorithm DOM finds the exact coloring*** (coloring with the minimum number of colors).

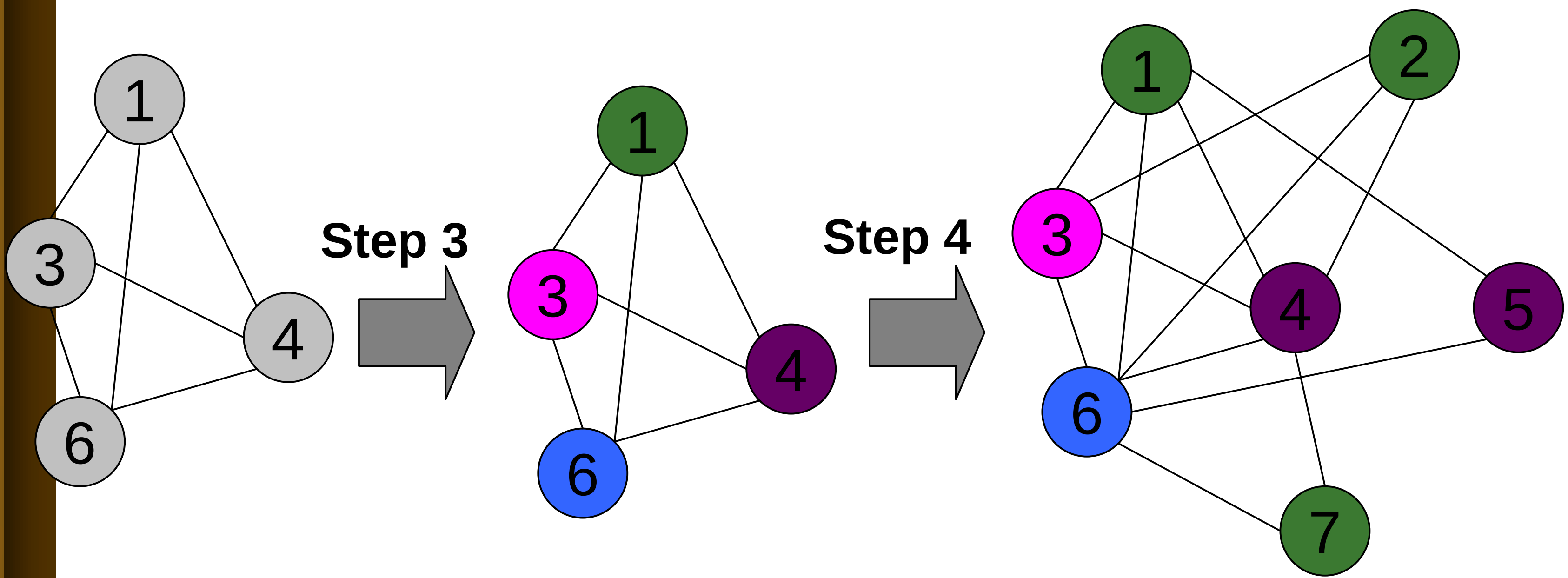
Example Showing How DOM Colors a Reducible Graph



Step 1: Removing 2 and 7 covered by 1

Step 2: Removing 5 covered by 4

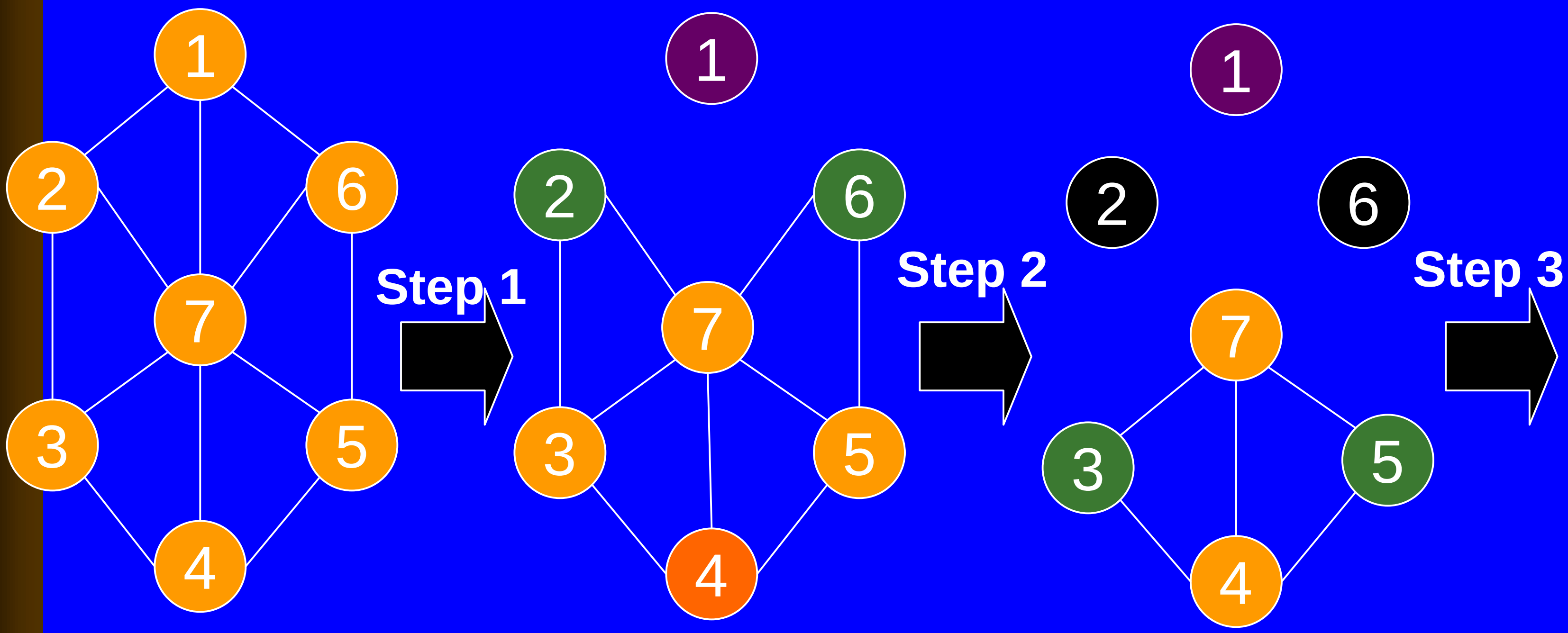
Example Showing How DOM Colors of a Reducible Graph



Step 3: Coloring the complete graph

Step 4: Coloring the covered vertices

Example Showing How DOM Colors of a Non-Reducible Graph

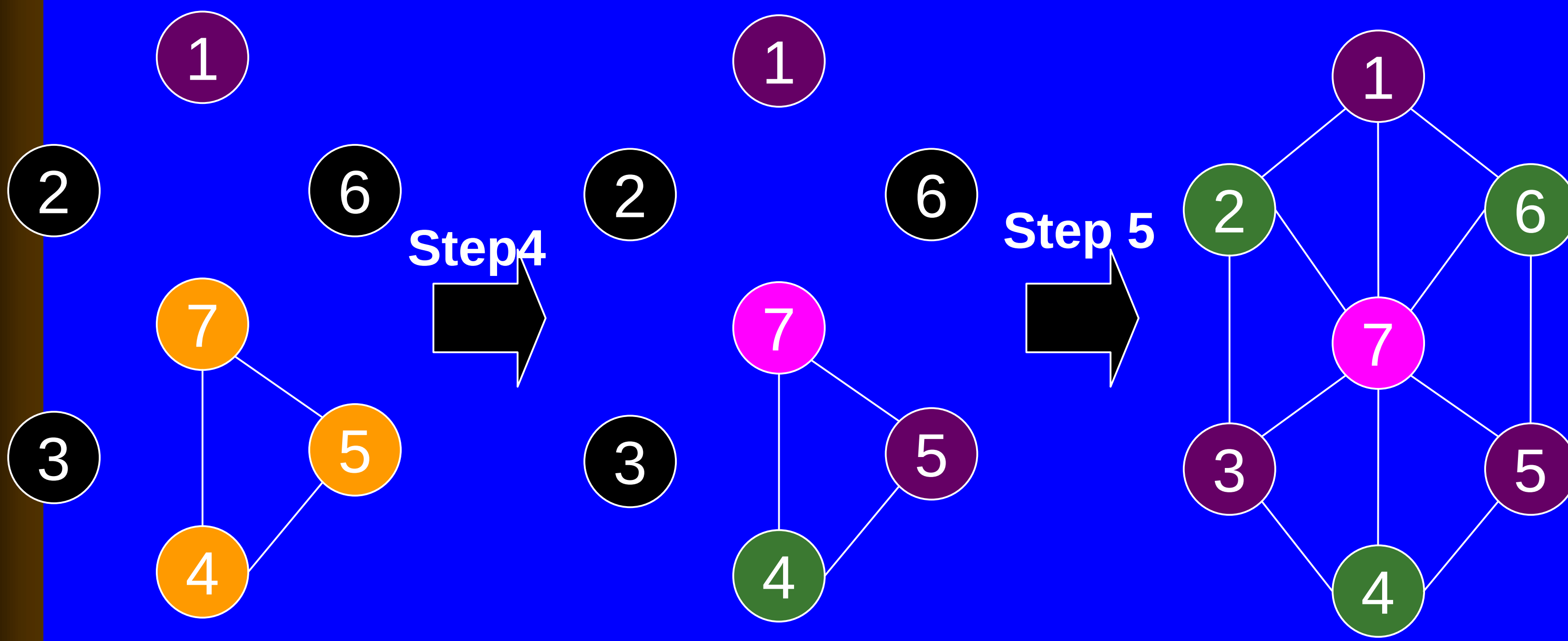


Step 1: Removing random node (1)

Step 2: Removing 2 and 6 covered by 4

Step 3: Removing 3 pseudo-covered by 5

Example Showing How DOM Colors of a Reducible Graph



Step 4: Coloring the complete graph

Step 5: Coloring the remaining nodes

Comparison of Results Obtained by MVGUD on MCNC Benchmarks

Bmk	i	o	c	Alg	C	a bl	AvE%	NP	TC	AC	T, s
5xpl	7	10	143	EXOC	344	17	63	28	123	4.4	2006
				CLIP	344	17		28	123	4.4	29.5
				DOM	344	17		28	123	4.4	29.9
9syml	9	1	158	EXOC	96	3	48.7	11	54	4.9	108
				CLIP	96	3		10	52	5.2	55.2
				DOM	64	3		11	54	4.9	47.3
b12	15	9	172	EXOC	284	25	15	130	389	3.0	87.0
				CLIP	284	25		132	387	2.9	57.1
				DOM	284	25		130	389	3.0	46.4
bw	5	28	97	EXOC	560	56	55	115	361	3.1	51.0
				CLIP	560	56		115	361	3.1	50.9
				DOM	560	56		115	361	3.1	48.7

Total Colors Found by DOM and CLIP vs. Colors Found by EXOC

Number of errors	DOM								CLIP							
	B = 2		B = 4		B = 5		Total		B = 2		B = 4		B = 5		Total	
	N	%	N	%	N	%	N	%	N	%	N	%	N	%	N	%
0 (exact)	46	100	45	97.8	41	89.1	132	95.6	32	66.1	20	43.5	14	30.5	66	47.8
1	-	-	1	2.1	3	6.5	4	2.8	8	17.4	13	28.3	11	23.9	33	23.9
2	-	-	-	-	1	2.1	1	0.7	4	8.6	5	10.8	12	26.1	21	15.2
3	-	-	-	-	-	-	-	-	1	2.1	8	17.4	3	6.5	12	8.7
4	-	-	-	-	1	2.2	1	0.7	1	2.1	-	-	3	6.5	4	2.8
5	-	-	-	-	-	-	-	-	-	-	-	-	2	4.3	2	1.4
6	-	-	-	-	-	-	-	-	-	-	-	-	1	2.1	1	0.7

Abbreviations

- **TR - TRADE, an earlier decomposer developed at Porland State University**
- **MI - MISII, a decomposer from UC, Berkeley**
- **St - a binary decomposer from Freiberg (Germany), Steinbach**
- **SC - MuloP-dc, a decomposer from Freiburg (Germany), Scholl**
- **LU - program Demain from Warsaw/Monash (Luba and Selvaraj)**
- **Js and Jh - systematic and heuristic strategies in a decompower from Jozwiak Technical University of Eindhoven (Jozwiak)**

Comparison of MVGUD with Other Decomposers

Benchmark		Cost for Various Decomposers *								
Name	i(o)	TR	MI	St	SC	LU	Js	Jh	MV	Time, s
5xpl	7/10	496	384	292	288 (9)	288 (9)	320 (20)	336 (21)	236	11.0
9sym	9/1	640	984	400	224 (7)	160 (5)			104	26.4
con1	7/2	80	68	60					70	2.3
duke2	22/29	6516	2428	2200	3456 (108)				2896	11289.0
ex5p	8/63		3720	1560					2104	208.0
f5lm	8/8	372	392	240	256 (8)				177	10.1
misex1	8/7	472	208	224	256 (8)	354 (11)	304 (19)	288 (18)	229	8.6
misex2	25/18	548	464	436	768 (24)				392	1086.0
misex3	14/14	9816	4204	3028					1744	1316.0

* Abbreviation are explained in the previous slide

Other Topics - Review

Definition of a Cartesian Product

Definition of a Relation as a subset of Cartesian Product

Oriented and non-oriented relations

Characteristic function of a relation

This to be covered only if students do not have background!

More on combinatorial problems

- Graph coloring applied to SOP minimization
- What is a relation and characteristic function
- coloring and other machines based on circuits
- satisfiability/Petrick machines

What have we learnt?

- Finding the minimum column multiplicity for a bound set of variables is an important problem in Curtis decomposition.
- We compared two graph-coloring programs: one exact, and other one based on heuristics, which can give, however, provably exact results on some types of graphs.
- These programs were incorporated into the multi-valued decomposer MVGUD, developed at Portland State University.

What have we learned (cont)

- We proved that the exact graph coloring is not necessary for high-quality decomposers.
- We improved by orders of magnitude the speed of the column multiplicity problem, with very little or no sacrifice of decomposition quality.
- Comparison of our experimental results with competing decomposers shows that for nearly all benchmarks our solutions are best and time is usually not too high.

What have we learnt (cont)

- Developed a new algorithm to create incompatibility graphs
- Presented a new heuristic dominance graph coloring program DOM
- Proved that exact graph coloring algorithm is not needed
- Introduced early filtering of decompositions
- Shown by comparison that this approach is faster and gives better decompositions

What you have to remember

- How to decompose any single or multiple output Boolean function or relation using both Ashenhurst and Curtis decomposition
- How to do the same for multi-valued function or relation
- How to color graphs efficiently and how to write a LISP program for coloring

References

- **Partitioning for two-level decompositions**
  - M.A.Perkowski, “A New Representation of Strongly Unspecified Switching Functions and Its Application to Multi-Level AND/OR/EXOR Synthesis”, Proc. RM ‘95 Work, 1995, pp.143-151
- **Our approach to decomposition**
  - M.A.Perkowski, M.Marek-Sadowska, L.Jozwiak, M.Nowicka, R.Malvi, Z.Wang, J.Zhang, “Decomposition of Multiple-Valued Relations”, Proc. ISMVL ‘97, pp.13-18

References

- **Our approach to graph coloring**
  - M.A.Perkowski,R.Malvi,S.Grygiel,M.Burns, A.Mishchenko,"Graph Coloring Algorithms for Fast Evaluation of Curtis Decompositions," Proc. Of Design Automation Conference, DAC'99, pp.225-230.