# Integer Programming Formulations of Graph Colouring



Jakub Marecek
IBM



Andrew J. Parkes
University of Nottingham

# Integer Programming Formulations of Graph Colouring

Jakub Mareček[1,2]
w/ Edmund K. Burke[1], Andrew J. Parkes[1] and Hana Rudová[2]

[1]Automated Scheduling, Optimisation and Planning
School of Computer Science and IT, The University of Nottingham
Wollaton Road, Nottingham NG8 1BB, UK
[2]Faculty of Informatics, Masaryk University
Botanická 68a, Brno 602 00, The Czech Republic
E-mail: jakub@marecek.cz

October 18, 2007

# Graphs, Cliques, Independent Sets, Colourings

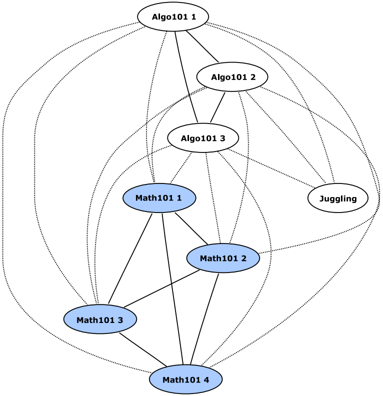
Graph $G = (V, E)$:

- is a set of vertices plus a set of two-element pairs of vertices
- thus is not oriented, without loops $\{u, u\} \in E$
- has vertices $u, v$ adjacent if they form an edge $\{u, v\} \in E$

## Graphs, Cliques, Independent Sets, Colourings

Subset $S$ of vertices of a graph $G = (V, E)$ is:

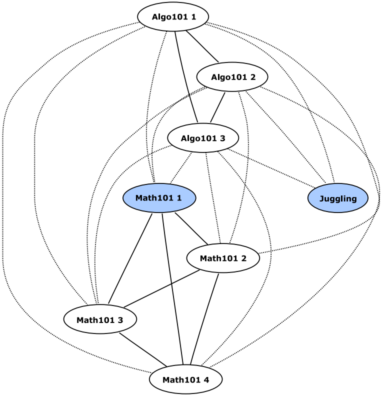
- clique, if each two $u, v \in S$ form an edge $\{u, v\} \in E$
- independent, if no two $u, v \in S$ form an edge $\{u, v\} \in E$
- dominating (max. independent), if each vertex in the graph is either in $S$ or is adjacent to some vertex in $S$



# Graphs, Cliques, Independent Sets, Colourings

Subset $S$ of vertices of a graph $G = (V, E)$ is:

- clique, if each two $u, v \in S$ form an edge $\{u, v\} \in E$
- independent, if no two $u, v \in S$ form an edge $\{u, v\} \in E$
- dominating (max. independent), if each vertex in the graph is either in $S$ or is adjacent to some vertex in $S$



# Graphs, Cliques, Independent Sets, Colourings

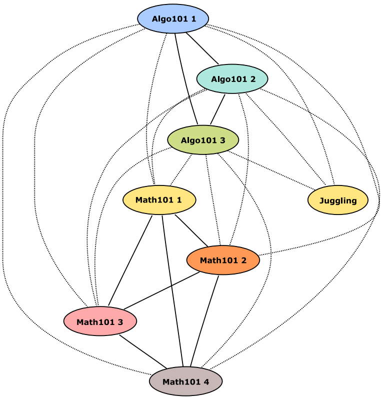
Colouring of $G = (V, E)$:

- assign colours $c \in \{1, 2, \ldots, k\}$ to vertices of $G = (V, E)$ such that each two adjacent vertices are assigned different $c$
- vertices of one colour form an independent set
- each edge and clique: an `all_different` constraint
- $\mathcal{NP}$-Complete
- benchmark by Johnson and Trick (1996)

# Graphs, Cliques, Independent Sets, Colourings

Colouring of $G = (V, E)$:

- assign colours $c \in \{1, 2, \ldots, k\}$ to vertices of $G = (V, E)$ such that each two adjacent vertices are assigned different $c$
- vertices of one colour form an independent set
- each edge and clique: an `all_different` constraint
- $\mathcal{NP}$-Complete
- benchmark by Johnson and Trick (1996)



# Integer Programming: A Quick Guide

1. Come up with an encoding of a solution
2. Come up with variables suitable for storing that encoding
3. Come up with a set of linear equations in those variables, which are satisfied if and only if variables encode a feasible solution
4. Optionally come up with some more linear equations ("cuts") that have to be satisfied for each feasible solution

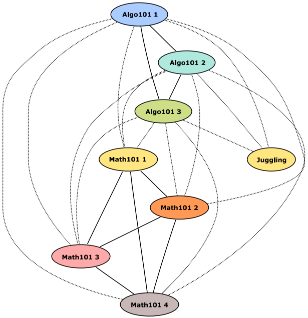
# Advanced Integer Programming: A Quick Guide

- "Exponential" spells trouble!
- Cut Generation ("Branch-and-Cut"): if you end up with an exponential no. of constraints, you have to figure out on the fly which are violated when and add/remove them in the process of solving
- Column Generation ("Branch-and-Price"): if you end up with an exponential no. of variables, you have to figure out which where important when and add/remove them in the process of solving

# The Standard Formulation

$$x_{v,c} = \begin{cases} 1 & \text{if vertex } v \text{ is coloured with colour } c \\ 0 & \text{otherwise} \end{cases}$$

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| Math101_1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Math101_2 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Math101_3 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Math101_4 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| Algo101_1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Algo101_2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| Algo101_3 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Juggling | 1 | 0 | 0 | 0 | 0 | 0 | 0 |



# The Standard Formulation

$$x_{v,c} = \begin{cases} 1 & \text{if vertex } v \text{ is coloured with colour } c \\ 0 & \text{otherwise} \end{cases}$$

$$\sum_{c=1}^{k} x_{v,c} = 1 \quad \forall \text{ vertices } v \in V$$

$$x_{u,c} + x_{v,c} \leq 1 \quad \forall \text{ colours } c \in K \quad \forall \text{ edges } \{u, v\} \in E$$

- $k\,|V|$ binary variables and $k\,|E|$ constraints
- Mehrotra and Trick (1996): imagine $x_{v,c} = 1/k \quad \forall v \forall c$
- Coll, Marenco, Méndez-Díaz, and Zabala (2002): polyhedron
- Zabala and Méndez-Díaz (2006): branch-and-cut

# Extension: Synchronisation with General Integer Variables

$X_v = c$ if colour $c$ used to colour vertex $v$

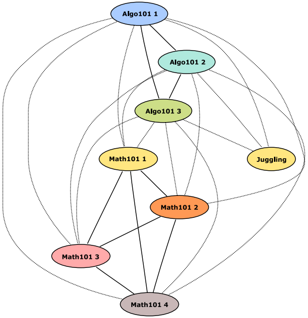
$$X_v - \sum_{c=1}^{k} c x_{v,c} = 0 \quad \forall \text{ vertices } v \in V$$

- $|V|$ additional variables and $|V|$ additional constraints
- Williams and Yan (2001): does not perform well

# The Binary Encoded Formulation

$$x_{v,b} = \begin{cases} 1 & \text{if vertex } v \text{ is assigned colour having bit } b \text{ set to } 1 \\ 0 & \text{otherwise} \end{cases}$$

| | 1 | 2 | 3 |
|---|---|---|---|
| Math101_1 | 1 | 0 | 0 |
| Math101_2 | 0 | 1 | 0 |
| Math101_3 | 1 | 1 | 0 |
| Math101_4 | 0 | 0 | 1 |
| Algo101_1 | 1 | 0 | 1 |
| Algo101_2 | 0 | 1 | 1 |
| Algo101_3 | 1 | 1 | 1 |
| Juggling | 1 | 0 | 0 |



# The Binary Encoded Formulation

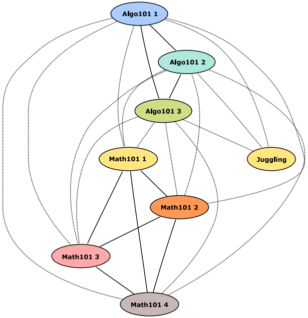
$$x_{v,b} = \begin{cases} 1 & \text{if vertex } v \text{ is assigned colour having bit } b \text{ set to } 1 \\ 0 & \text{otherwise} \end{cases}$$

- $\lceil \log_2 k \rceil \, |V|$ variables
- Lee (2002; 2007): the `all_different` polyhedron
- three exp. large classes of inequalities
- perhaps suitable for edge colouring?

# An Independent Set Formulations

$$x_i = \begin{cases} 1 & \text{if dominating set } i \text{ is assigned a single colour} \\ 0 & \text{otherwise} \end{cases}$$

| | used? |
|---|---|
| Algo101_1 | 1 |
| Algo101_2 | 1 |
| Algo101_3 | 1 |
| Math101_1 - Juggling | 1 |
| Math101_2 - Juggling | 1 |
| Math101_3 - Juggling | 1 |
| Math101_4 - Juggling | 1 |



# An Independent Set Formulations

$$x_i = \begin{cases} 1 & \text{if dominating set } i \text{ is assigned a single colour} \\ 0 & \text{otherwise} \end{cases}$$

$$\sum_{i \in I} x_i \leq k$$

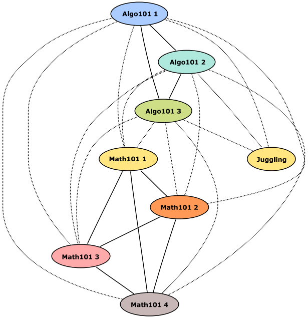
$$\sum_{i \in I,\ \text{s.t.}\ v \in i} x_i \geq 1 \quad \forall \text{ vertices } v \in V$$

- Mehrotra and Trick (1996): the first alternative formulation
- based on set $I$ of maximal independent sets
- $|V| + 1$ constraints + exp. no. of variables
- Column generation, post-processing

# Another Independent Set Formulation

$$x_i = \begin{cases} 1 & \text{if independent set } i \text{ is assigned a single colour} \\ 0 & \text{otherwise} \end{cases}$$

| | used? |
|---|---|
| Math101_1 | 0 |
| Math101_2 | 1 |
| Math101_3 | 1 |
| Math101_4 | 1 |
| Algo101_1 | 1 |
| Algo101_2 | 1 |
| Algo101_3 | 1 |
| Juggling | 0 |
| Math101_1 - Juggling | 1 |
| Math101_2 - Juggling | 0 |
| Math101_3 - Juggling | 0 |
| Math101_4 - Juggling | 0 |



# Another Independent Set Formulation

$$x_i = \begin{cases} 1 & \text{if independent set } i \text{ is assigned a single colour} \\ 0 & \text{otherwise} \end{cases}$$

$$\sum_{i \in I} x_i \leq k$$

$$\sum_{i \in I, \text{ s.t. } v \in i} x_i = 1 \quad \forall \text{ vertices } v \in V$$

- Mehrotra and Trick (1996), Hansen, Labbé, and Schindl (2005)
- based on set $I$ of (not necessarily max.) independent sets
- $|V| + 1$ constraints + exp. no. of variables
- Column generation

# A Scheduling Formulation (with Precedency Constraints)

$$X_v \quad = c \text{ if colour } c \text{ used to colour vertex } v$$

$$x_{u,v} = \begin{cases} \perp & \text{if vertex } u = v \\ 1 & \text{if vertices } u, v \text{ are assigned colours } c_u, c_v \text{ with } c_u < c_v \\ 0 & \text{otherwise} \end{cases}$$

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| Math101_1 | | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| Math101_2 | 0 | | 1 | 1 | 1 | 1 | 1 | 0 |
| Math101_3 | 0 | 0 | | 1 | 1 | 1 | 1 | 0 |
| Math101_4 | 0 | 0 | 0 | | 1 | 1 | 1 | 0 |
| Algo101_1 | 0 | 0 | 0 | 0 | | 1 | 1 | 0 |
| Algo101_2 | 0 | 0 | 0 | 0 | 0 | | 1 | 0 |
| Algo101_3 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 |
| Juggling | 0 | 1 | 1 | 1 | 1 | 1 | 1 | |

# A Scheduling Formulation (with Precedency Constraints)

$$X_v \quad = c \text{ if colour } c \text{ used to colour vertex } v$$

$$x_{u,v} = \begin{cases} \perp & \text{if vertex } u = v \\ 1 & \text{if vertices } u, v \text{ are assigned colours } c_u, c_v \text{ with } c_u < c_v \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{aligned} X_u - X_v - kx_{u,v} &\leq -1 && \forall \text{ edges } \{u, v\} \in E \\ X_v - X_u - kx_{u,v} &\leq k - 1 && \forall \text{ edges } \{u, v\} \in E \end{aligned}$$

- $|V|\,(|V| - 1)$ variables and $2\,|E|$ precedency constraints
- Williams and Yan (2001): detailed study, “poor relaxations”
- Branch and cut

# Encoding Using Acyclic Orientations

## Definition

An acyclic orientation $G' = (V, E')$ of an undirected $G = (V, E)$ is a directed graph such that for each $\{u, v\} \in E$, there is either $(u, v) \in E'$ or $(v, u) \in E'$, and there is no directed cycle in $G'$.

## Theorem

*Deming (1979): if $\chi$ is the smallest $k$, such that there is a $k$-colouring of $G$, there is an acyclic orientation of $G$, with longest path of $\chi$ vertices*

- Here be the lions: Never implemented.

# Formulation Using Asymmetric Representatives

$$x_{u,v} = \begin{cases} \perp & \text{if } u = v \text{ or if } \{u, v\} \in E \\ 1 & \text{if vertex } u \text{ represents the colour assigned also to vertex } v \\ 0 & \text{otherwise} \end{cases}$$

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| Math101_1 | 0 | | | | | | | 1 |
| Math101_2 | | 1 | | | | | | 0 |
| Math101_3 | | | 1 | | | | | 0 |
| Math101_4 | | | | 1 | | | | 0 |
| Algo101_1 | | | | | 1 | | | |
| Algo101_2 | | | | | | 1 | | |
| Algo101_3 | | | | | | | 1 | |
| Juggling | 0 | 0 | 0 | 0 | | | | 0 |

# Formulation Using Asymmetric Representatives

$$x_{u,v} = \begin{cases} \perp & \text{if } u = v \text{ or if } \{u, v\} \in E \\ 1 & \text{if vertex } u \text{ represents the colour assigned also to vertex } v \\ 0 & \text{otherwise} \end{cases}$$

- $|V| + |V|^2 - |E|$ variables and $\mathcal{O}(|V|^3)$ constraints
- Campêlo, Campos, and Corrêa (2007): "representatives"
- Campêlo et al. (2007): order vertices, which induces an acyclic orientation, add symmetry-breaking constraints
- No empirical results are available

# A Summary

Known integer programming formulations of graph colouring:

| Based on | Pre-proc. | Variables | Constraints | Post-proc. |
|---|---|---|---|---|
| Vertices | Easy | $k\,\lvert V\rvert$ | $k\,\lvert E\rvert$ | Easy |
| Max. Independent | Hard | Exp. many | $\lvert V\rvert + 1$ | Hard |
| Any Independent | Hard | Exp. many | $\lvert V\rvert + 1$ | Easy |
| Binary Encoding | Easy | $\lceil \log_2 k \rceil\,\lvert V\rvert$ | Exp. many | Easy |
| Precedencies | Easy | $\lvert V\rvert^2$ | $2\,\lvert E\rvert$ | Easy |
| Ac. Orientations | Easy | $\lvert E\rvert$ | Exp. many | Hard |
| Asymmetric Reps. | Easy | $\mathcal{O}(\lvert E\rvert)$ | $\mathcal{O}(\lvert V\rvert\,\lvert E\rvert)$ | Easy |

What is good? What is bad?

# What about something new?

1. Come up with an encoding of a solution
2. Come up with variables suitable for storing that encoding
3. Come up with a set of linear equations in those variables, which are satisfied if and only if variables encode a feasible solution
4. Optionally come up with some more linear equations ("cuts") that have to be satisfied for each feasible solution

# What about something new?

## Definition

Clique partition of graph $G = (V, E)$ is a pair $(Q, E')$, where $Q$ is a partition of vertices $V$, such that for all sets $q \in Q$, all $v \in Q$ are pairwise adjacent in $G$, and
$E' = \{\{q_u, q_v\} | \{u, v\} \in E, q_u, q_v \in Q, q_u \neq q_v, u \in q_u, v \in q_v, \}$.

- For general graphs, it's $\mathcal{NP}$-Hard to find a clique partition of minimum cardinality (of $Q$)
- In a number of applications, a (not min.) partition of the vertex set into (not max.) cliques is given implicitly
- In Udine CTT, it's "events grouped by the course"
- It should be possible to take advantage of this

# What about something new?

**Definition**

Good clique partition $(Q, E')$ of a graph $G = (V, E)$ maitains the following property: if there exists $\{q_u, q_v\} \in E'$, then for all $u \in q_u$ and for all $v \in q_v$, there exists an edge $\{u, v\} \in E$.

- What is the structure of the clique partition?
- In Udine CTT, if there is a student enrolled in courses $c$ and $d$, there are edges connecting each event of $c$ with each event of $d$
- The good clique partition is also a "suitable" clique cover

# What about something new?

**Definition**

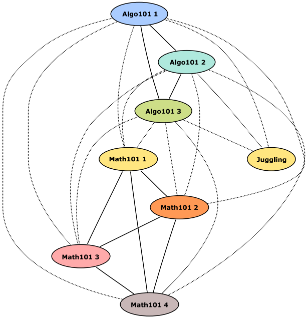
Set colouring of a graph $G = (V, E)$ assigning each vertex $f : V \rightarrow \mathbb{N}$ colours out of the set $K = \{1, \ldots, k\}$, is a mapping $c : V \rightarrow 2^K$, such that for all $v \in V : |c(v)| = f(v)$ and for all $\{u, v\} \in E$, $c(u) \cap c(v) = \emptyset$.

- It's $\mathcal{NP}$-Hard to find a minimum cardinality clique cover
- Given a good clique partition, it remains $\mathcal{NP}$-Complete to decide, if there exists a set colouring of $G'$ with $f(q)$ using $k$ colours
- If we have one, we can reformulate vertex colouring as set colouring
- Assigning each vertex a set of colours of cardinality equal to the size of the clique it represents

# A New Clique-Based Formulation

$$x_{q,c} = \begin{cases} 1 & \text{if colour } c \in K \text{ is included in the set assigned to } q \in Q \\ 0 & \text{otherwise} \end{cases}$$

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| Math101 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| Algo101 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| Juggling | 1 | 0 | 0 | 0 | 0 | 0 | 0 |



# A New Clique-Based Formulation

$$x_{q,c} = \begin{cases} 1 & \text{if colour } c \in K \text{ is included in the set assigned to } q \in Q \\ 0 & \text{otherwise} \end{cases}$$

$$\sum_{c=1}^{k} x_{q,c} = f(q) \quad \forall \text{ vertices } q \in Q$$

$$x_{u',c} + x_{v',c} \leq 1 \ \forall \quad \text{colours } c \in K \quad \forall \text{ edges } \{u', v'\} \in E'$$

- $k\,|Q|$ binary variables and $|Q| + k\,|E'|$ constraints

## How good is it?

- Breaks some symmetries
- For a trivial integer programming solver (w/o bounding & cuts) speed-up by the factor of:

$$\prod_{q \in Q} |q|!$$

- For a modern IP solver, speed-up by the factor of at least

$$|V| \;/\; |Q|$$

- There are $|V| - |Q|$ fewer variables, without raising the number of constraints or making the constraint matrix considerably denser

# Empirical Tests

- Public (7) and not-yet-public (7) instances from Udine
- Own rule-based generator of random instances
- Roughly ($e/10$) curricula, ($e/6$) teachers, ($e/3$) courses

- CPLEX 10.0 on Faramir
- Default parameters, symmetry-breaking off, 1hr time limit

# Empirical Results I

Performance on colouring (feasibility):

| Inst. | Std. | Its. | New | Its. | $\frac{\text{Std.}}{\text{New}}$ |
|---|---|---|---|---|---|
| rand01 | 2.85s | 1635 | 0.90s | 931 | 3.16 |
| rand02 | 2.99s | 1666 | 0.94s | 1106 | 3.18 |
| rand03 | 9.92s | 5792 | 1.05s | 1045 | 9.45 |
| rand04 | 99.48s | 26317 | 5.18s | 2802 | 19.20 |
| rand05 | 73.72s | 19802 | 33.49s | 17467 | 2.20 |
| rand06 | 83.78s | 22537 | 40.35s | 19836 | 2.08 |
| rand07 | 216.08s | 35821 | 86.44s | 25541 | 2.50 |
| rand08 | 59.70s | 10760 | 43.45s | 13342 | 1.37 |
| rand09 | 127.19s | 22155 | 98.32s | 25782 | 1.29 |
| rand11 | 3.80s | 1761 | 1.51s | 1194 | 2.52 |
| rand12 | 4.55s | 2005 | 2.31s | 1377 | 1.97 |

# Empirical Results II

| | | | | | |
|---|---|---|---|---|---|
| rand13 | 95.67s | 22851 | 47.94s | 18957 | 2.00 |
| rand14 | 45.25s | 10544 | 6.64s | 2629 | 6.81 |
| rand15 | 30.77s | 6799 | 6.89s | 2685 | 4.47 |
| rand16 | 114.32s | 11603 | 275.44s | 51518 | 0.42 |
| rand17 | 251.15s | 33185 | 144.93s | 36949 | 1.73 |
| rand18 | 160.25s | 21686 | 138.04s | 34461 | 1.16 |
| udine1 | 23.23s | 8082 | 4.45s | 3370 | 5.22 |
| udine2 | 14.51s | 4749 | 10.04s | 4826 | 1.45 |
| udine3 | 83.41s | 16807 | 17.25s | 11698 | 4.84 |
| udine4 | 144.49s | 30655 | 145.99s | 30655 | 0.99 |

# Empirical Results III

Performance on Udine CTT:

| Inst. | Std. | Its. | New | Its. | $\frac{\text{Std.}}{\text{New}}$ |
|---|---|---|---|---|---|
| rand01 | 385.59s | 180854 | 84.42s | 43737 | 4.57 |
| rand02 | 290.09s | 71537 | 72.42s | 34296 | 4.01 |
| rand03 | 443.95s | 148961 | 59.99s | 23310 | 7.40 |
| rand04 | gap 0.24% | 419910 | 1242.50s | 210104 | |
| rand05 | gap 4.15% | 360868 | 1194.71s | 250148 | |
| rand06 | gap 8.33% | 299998 | 1257.72s | 247075 | |
| rand07 | gap 89.71% | 234087 | gap 90.11% | 242978 | |
| rand08 | gap 99.85% | 237243 | gap 99.90% | 312158 | |
| rand09 | gap 93.97% | 199619 | gap 95.44% | 263820 | |
| rand10 | 285.91s | 66842 | 70.17s | 27416 | 4.07 |
| rand11 | 211.71s | 68244 | 61.32s | 31738 | 3.45 |
| rand12 | 337.31s | 129788 | 84.16s | 48401 | 4.01 |

# Empirical Results IV

| | | | | | |
|---|---|---|---|---|---|
| rand13 | gap 0.24% | 431148 | 884.60s | 175513 | |
| rand14 | gap 6.47% | 322073 | 1356.97s | 320129 | |
| rand15 | gap 1.74% | 303518 | 1166.50s | 280722 | |
| rand16 | gap 66.44% | 175766 | gap 67.19% | 417706 | |
| rand17 | gap 94.15% | 239576 | gap 94.06% | 293519 | |
| rand18 | gap 90.57% | 251822 | gap 49.34% | 345817 | |
| udine1 | 1175.40s | 166539 | 237.12s | 104221 | 4.96 |
| udine2 | gap 100.00% | 639068 | gap 100.00% | 3318838 | |
| udine3 | gap 99.31% | 367505 | gap 59.59% | 2000062 | |
| udine4 | gap 99.69% | 220364 | gap infinite | 962856 | |

# Conclusions

Integer programming formulations of graph colouring:

| Based on | Pre-proc. | Variables | Constraints | Post-proc. |
|---|---|---|---|---|
| Vertices | Easy | $k\|V\|$ | $k\|E\|$ | Easy |
| Max. Independent | Hard | Exp. many | $\|V\|+1$ | Hard |
| Any Independent | Hard | Exp. many | $\|V\|+1$ | Easy |
| Binary Encoding | Easy | $\lceil \log_2 k \rceil \|V\|$ | Exp. many | Easy |
| Precedencies | Easy | $\|V\|^2$ | $2\|E\|$ | Easy |
| Ac. Orientations | Easy | $\|E\|$ | Exp. many | Hard |
| Asymmetric Reps. | Easy | $\mathcal{O}(\|E\|)$ | $\mathcal{O}(\|V\|\|E\|)$ | Easy |
| Cliques | Hard | $k\|Q\|$ | $\|Q\|+k\|E'\|$ | Easy |
| Cliques (Udine) | Easy | $k\|Q\|$ | $\|Q\|+k\|E'\|$ | Easy |

The $1M Question: Which one is the best?

# Conclusions

- The answer: depends on soft constraints
- Our formulation works well for some timetabling apps[1] and does not require column generation ...
- ... but cannot beat good column generation in general (e.g. dense random graphs)
- Overall, integer programming is a great tool

[1] Where you are given a clique partition.

# Questions are Welcome!

- Any questions or comments are very welcome!

- See `http://cs.nott.cz/~jxm` for more
- Come and talk to me in B78
- We have a draft and love to get feedback

# Bibliography I

# Bibliography II

# Bibliography III

Campêlo, M., Campos, V. A., & Corrêa, R. C. (2007). On the asymmetric representatives formulation for the vertex coloring problem. *Disc. App. Math.*, in press.

Coll, P., Marenco, J., Méndez-Díaz, I., & Zabala, P. (2002). Facets of the graph coloring polytope. *Ann. Op. Res.*, *116*, 79–90.

Deming, R. W. (1979). Acyclic orientations of a graph and chromatic and independence numbers. *J. Comb. Theory, Ser. B*, *26*(1), 101–110.

Hansen, P., Labbé, M., & Schindl, D. (2005). *Set covering and packing formulations of graph coloring: algorithms and first polyhedral results* (Tech. Rep. No. G-2005-76). Montreal, Canada: GERAD.

# Bibliography IV

Johnson, D. J., & Trick, M. A. (Eds.). (1996). *Cliques, coloring, and satisfiability: Second dimacs implementation challenge, workshop, october 11-13, 1993*. Boston, MA, USA: American Mathematical Society.

Lee, J. (2002). All-different polytopes. *J. Comb. Optim.*, *6*(3), 335–352.

Lee, J., & Margot, F. (2007). On a binary-encoded ILP coloring formulation. *INFORMS J. Computing*, *19*(3), 406–415.

Mehrotra, A., & Trick, M. A. (1996). A column generation approach for graph coloring. *INFORMS J. Computing*, *8*(4), 344–354.

Williams, H. P., & Yan, H. (2001). Representations of the all_different predicate of constraint satisfaction in integer programming. *INFORMS J. Computing*, *13*(2), 96–103.

# Bibliography V

Zabala, P., & Méndez-Díaz, I. (2006). A branch-and-cut algorithm for graph coloring. *Disc. App. Math.*, *154*(5), 826–847.