Matheuristic Variants of DSATUR for the Vertex Coloring Problem

Federico Bruzzone,[1] PhD Candidate

Milan, Italy – 4 December 2025

[1]ADAPT Lab – Università degli Studi di Milano,
Website: federicobruzzone.github.io,
Github: github.com/FedericoBruzzone,
Email: federico.bruzzone@unimi.it
Slides: TODO

Notation

- $G = (V, E)$ is an undirected graph with vertex set $V = \{v_1, ..., v_n\}$ and edge set E, and denote $I = [\![1; n]\!] = [1; n] \cap \mathbb{Z}$.
- An edge $e = (v_i, v_j) \in E$ with $i < j$ links the underlying vertices (*for* ***VCP*** *there is no sense to consider loop/multiple edges*).
- For $i \in I$, δ_i denotes the set of neighbors of vertex v_i, and $d_i = |\delta_i| = \{j \in I \mid \mathrm{ngb}(v_i, v_j) = 1\}$, where $\mathrm{ngb}(v_i, v_j) = 1\ iif\ (v_i, v_j) \in E$.[2]

[2] ngb stands for "neighbor".

A k-coloring of G

- It is an assignment of colors to vertices such that no two adjacent vertices share the same color.
- **VCP** consists in finding a k-coloring of G using the minimum number of colors k (the *chromatic number* $\chi(G)$).
- A valid k-coloring (c) fulfills: $\forall i < j, \left(v_i, v_j\right) \in E \Longrightarrow c_i \neq c_j$ where $c_i = [\![1; k]\!]$ is the color of v_i.

A partial k-coloring of G

- If a vertex may not be colored, we set $c_i = -1 \; s.t. \; c_i \in [\![1; k]\!] \cup \{-1\}$
- A partial k-coloring (c) is *feasible* if $\forall i < j, (v_i, v_j) \in E \Longrightarrow c_i \neq c_j \vee c_i = c_j = -1$.
- Given v_i the saturation table S_i is the set of colors assigned to its colored neighbors: $S_i = \bigcup_{j \in \delta_i}^{n} \{c_j\} \setminus \{-1\}$, and $s_i = |S_i|$ is the saturation degree.
- A total order $\succcurlyeq$ over V is defined as: $v_i \succcurlyeq v_j \Longleftrightarrow s_i > s_j \vee (s_i = s_j \wedge d_i \geq d_j)$

Compact ILP Formulations[3], feasible *iif* $\chi(G) \leq k$

$z_{i,c} \in \{0,1\}$ indicates if vertex v_i is assigned color c.

$y_c \in \{0,1\}$ indicates if color c is used in the coloring.

$$\min \sum_{c=1}^{k} y_c$$

$$s.t. \min \sum_{c=1}^{k} z_{i,c} = 1 \quad \forall i \in I$$

$$z_{i,c} + z_{j,c} \leq y_c \quad \forall (v_i, v_j) \in E, \forall c \in [\![1;k]\!]$$

The objective minimizes the number of used colors.

The 1st set ensures that each vertex is assigned exactly one color.

The 2nd set ensures that adjacent vertices do not share the same color.

[3]Efficient formulations: extended column generation by Furini and Malaguti [1], and reduced formulation to MWSSP by Cornaz et al. [2].

Observations

- Having an upper bound of the chromatic number as the initial value k (or simply $k = |V|$) guarantees the optimality of the solution.
- The size of k strongly affects the performance of ILP solvers.
- Symmetries in the model (e.g., colors are permutable) enlarge the search space for Branch-and-Bound algorithms (the same solution can be represented in multiple ways)

Representative ILP Model[4], asymmetric and easily to LP-relax

$x_{i,i'} \in \{0,1\}$, $\forall i, i' \in V s.t. i \le i'$, indicates if verticies v_i and $v_{i'}$ share the same color and i is the minimum index of its color class.

$$\min_z \sum_{i=1}^{n} x_{i,i}$$

$$s.t. \sum_{i' \le i} x_{i',i} \ge 1 \qquad \forall i \in I$$

$$x_{j,i} + x_{i,j} \le x_{j,j} \qquad \forall (v_i, v_j) \in E, \forall i < j \in I$$

The objective counts the number of representative vertices (i.e., used colors).

The 1st set ensures either $x_{i,i} = 1$ or v_i shares the color with a vertex $v_{i'}$ with $i' < i$.

The 2nd set expresses the color incompatibility between adjacent vertices and $x_{j,i} = 1$ implies that $x_{j,j} = 1$

[4] A vertex is representative of its color class if it has the minimum index among the vertices sharing the same color.

Standard DSATUR Algorithm

Algorithm 1: Standard DSATUR algorithm

Input: $G = (V, E)$ a non-empty and non-oriented graph
Initialization:
  define partial coloring c with $c_i := -1$ for all $i \in I$
  define saturation table S with $S_i := \emptyset$ for all $i \in I$
  initialize set $U := V$, and color $k := 0$
  while $U \neq \emptyset$
    find $u \in U$, a maximum of $\succcurlyeq$ in U.
    if $|S_u| = k$ **then** $k := k + 1$ // a new color is added
    compute $c_i := \min S_u$ // assign color to u
    remove u from U
    for all $i \in \delta_u \cap U$, $S_i = S_i \cup \{c_i\}$ // update saturation
  end while
return color k and (c) a k-coloring of G

- DSATUR is an adaptive greedy heuristic proposed by Brélaz [3], which colors vertices iteratively.
- Selection of the uncolored verex to color is given with order $\succcurlyeq$, maximizing first the saturation degree and secondly the degree.
- Coloring a new vertex updates saturation, the iteration order of vertices is thus adaptive.

DASTUR Matheuristic Variants[5]

[5] N. Dupin, "Matheuristic Variants of DSATUR for the Vertex Coloring Problem," in *Metaheuristics* 2024 [4]

Initialization

Defining an initial partial coloring and computing the saturation table for the uncolored vertices, **before** starting the main DSATUR iterations.

Variants:

1. `maxDeg`: color the vertex with the maximum degree — equivalent to standard DSATUR by definition of $\succcurlyeq$, it would suffer from many ties;
2. `col-`n: cosider n vertices having the maximum degree and color them solving a representative ILP model for the *induced* subgraph — more depth pre-processing, it tries to prevent erroneous decisions in the initial steps of DSATUR;
3. `clq`: find a maximum clique[6] and color it with different colors — an exact pre-processing (not heuristic), it leads to a better initial saturation table S for the uncolored vertices;
4. `clq-col-`n: combine `clq` and `col-`n — best of both worlds.

---

[6] It is NP-hard, an heuristic can be used.

Local Optimization with Larger Neighborhoods

(c) is a partial k-coloring, where k is the number of colors used until now.

- $C = \{i \in I \mid c_i > 0\}$ is the set of colored vertices in (c).
- $U \subset \{i \in I \mid c_i = -1\}$ is a subset of uncolored vertices in (c).

We want to define an ILP formulation to assign a color to each vertex $u \in U$ while preserving the colors of vertices in C.

Thank You!

Bibliography

[1] F. Furini and E. Malaguti, "Exact weighted vertex coloring via branch-and-price," *Discrete Optimization*, vol. 9, no. 2, pp. 130–136, 2012.

[2] D. Cornaz, F. Furini, and E. Malaguti, "Solving vertex coloring problems as maximum weight stable set problems," *Discrete Applied Mathematics*, vol. 217, pp. 151–162, 2017.

[3] D. Brélaz, "New methods to color the vertices of a graph," *Commun. ACM*, vol. 22, no. 4, pp. 251–256, Apr. 1979.

[4] N. Dupin, "Matheuristic Variants of DSATUR for the Vertex Coloring Problem," in *Metaheuristics*, M. Sevaux, A.-L. Olteanu, E. G. Pardo, A. Sifaleras, and S. Makboul, Eds., Cham: Springer Nature Switzerland, 2024, pp. 96–111.