International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(21s) 2026 pp. 717-730
https://doi.org/10.70917/ijcisim-2026-5339
Article

Graph Coloring Algorithms and Their Applications in Combinatorial Optimization: A Survey

Jisha Ann Abraham∗[1], C. Bazil Wilfred†[1], Thomaskutty Stephen‡[2]

[1]Department of Mathematics, Karunya Institute of Technology and Sciences, Karunya Nagar, Coimbatore 641114, Tamil Nadu, India.
[2]Department of Mathematics, Saintgits College of Engineering (Autonomous), Kottayam, Kerala, India
∗Corresponding author: jishaann@karunya.edu.in
†wilfbaz@karunya.edu
‡thomaskutty.stephen@saintgits.org

Abstract: Coloring the vertices, edges or faces of a graph so that no two adjacent elements share a label is among the oldest problems in graph theory, and one of the few whose reach extends into exam timetables and wireless spectrum allocation as it does into pure combinatorics. This survey draws together the problem's theoretical core – vertex, edge, face, list and total coloring – with the algorithms built to solve it and the industries that now depend on those algorithms. Because coloring is NP-hard, we trace the field's progression from exact and greedy methods (Welsh-Powell, DSATUR, backtracking) through metaheuristics that trade optimality for scale (genetic algorithms, tabu search, simulated annealing, ant colony and artificial bee colony optimization), to the graph neural network and quantum-inspired solvers that have emerged in recent years. Rather than treating theory, algorithms and applications as separate literatures, we connect them directly: each application – examination and crew scheduling, frequency assignment, compiler register allocation, cartographic map coloring – is traced back to the specific coloring variant and algorithm family the literature actually uses, while the algorithm families themselves are compared head-to-head on complexity, solution quality and scalability rather than catalogued one at a time. This comparative structure, together with its coverage of recent learning-based solvers, is what distinguishes this survey from the standard references on the subject. We close by outlining where the field's open problems remain, from long standing conjectures to the still-unanswered question of whether learned heuristics can match classical methods at real-world scale.
Keywords: graph coloring; vertex and edge coloring; NP-hard problems; metaheuristic algo-rithms; scheduling applications

1. Introduction

Graph coloring is a fundamental problem in graph theory concerned with assigning labels or colors, to the elements of a graph subject to adjacency constraints. In its most familiar form, vertex coloring, no two vertices joined by an edge may receive the same color, and the minimum number of colors needed to achieve this is the graph's chromatic number, $\chi(G)$. The problem admits several related formulations: edge coloring requires that no two edges sharing a vertex receive the same color; face coloring extends the constraint to the regions of a planar graph, with the Four-Color Theorem establishing that four colors always suffice [1, 2]; and total coloring unifies both vertex and edge constraints within a single coloring rule. Each variant has both a well-developed theory and a distinct set of computational techniques.

Determining a graph's chromatic number is NP-hard [25] and total coloring turns out to be no easier. That intractability has pushed the algorithmic literature in two different directions. The first sticks with classical, deterministic methods that give up optimality in exchange for speed. The Greedy algorithm is the simplest of these,

coloring vertices one at a time in whatever order they're given. Welsh and Powell improved on that by ordering vertices by decreasing degree before coloring [47] and Brélaz's DSATUR algorithm went a step further still, choosing at each step whichever vertex is adjacent to the most distinct colors already used [9]. All three run in low-order polynomial time, which is exactly why they have stayed in wide use — none of them guarantees an optimal coloring, but for many practical purposes that trade-off is worth making.

Where classical heuristics and exact methods such as backtracking reach their limits, a second body of work turns to metaheuristics, developed specifically to handle the instances the former cannot reasonably address. Genetic algorithms draw on principles of natural evolution, evolving a population of candidate colorings toward fewer conflicts through selection, crossover, and mutation [17]. Tabu search takes a different approach, operating from a single solution rather than a population and maintaining a memory of recently visited moves to avoid cycling back on itself [24]. Simulated annealing differs again as it deliberately accepts temporarily worse solutions according to a cooling schedule, allowing it to escape local optima rather than settle prematurely on the first solution it encounters. Ant colony and artificial bee colony optimization depart from all three, deriving their search strategies from collective biological behavior observed in insect colonies [43]. None of these methods offers an optimality guarantee, but each has been shown repeatedly to produce high-quality colorings on large, structurally complex graphs within

practical time limits.

Graph coloring's significance, however, is by no means confined to theory. In academic scheduling, it produces conflict-free examination and class timetables [34]. In telecommunica-tions, the same underlying formulation resolves frequency-assignment problems, keeping neigh-boring transmitters from interfering with one another [38]. Compiler design draws on it as well, where it determines how registers are allocated [11]. And in operations research, it supports both workforce and crew scheduling [21] and cartographic map coloring [50]. What these appli-cations share, despite arising from genuinely disparate fields, is that each reduces to the same underlying combinatorial structure.

Graph coloring already has several authoritative treatments — Jensen and Toft's *Graph*

Coloring Problems [25], Kubale's *Graph Colorings* [27] and Lewis's *Guide to Graph Colouring*

[30] chief among them — and these remain the field's standard theoretical references. This survey isn't trying to replace them. What it offers instead is narrower and more specific. Where those works present classical and metaheuristic algorithm families largely on their own terms, this paper compares them directly against one another on complexity, solution quality and scalability. Where the applications literature tends to sit apart from the algorithmic literature, this survey traces each major application domain back to the specific coloring variant and algorithm family it actually draws on. And where the standard references predate the recent turn toward learning, this paper brings in the graph neural network and quantum-inspired approaches to coloring that have emerged since 2022. The contribution here, in short, is that comparative synthesis and current coverage — not a new theoretical result.

The rest of the paper follows that structure. Section 2 lays out the methodology behind this review. Section 3 works through the fundamental definitions and classical theorems the later sections build on. From there, Section 4 turns to algorithms, moving from classical heuristics through metaheuristics to the learning-based methods that have emerged more recently, with comparative tables pulling their trade-offs together side by side. Section 5 then takes those same algorithm families and maps them onto the application domains where the literature actually puts them to use. The paper closes with Section 6, which sets out the open problems and directions still worth pursuing and Section 7, which draws the whole survey together.

2. Review Methodology

This paper is best understood as a narrative review rather than a formal systematic review conducted under a fixed protocol. The literature underlying were gathered through searches of Scopus, Web of Science, IEEE Xplore and Google Scholar, using a set of core terms — "graph coloring," "chromatic number," "graph coloring algorithm,"

"metaheuristic" and "graph neural network coloring" — supplemented by application-specific terms such as "frequency assignment," "exam timetabling" and "register allocation."

Older foundational works were included regardless of publication date, on the reasoning that results such as DSATUR, Vizing's theorem and Brooks' theorem remain in active use today, decades after they first appeared. The algorithmic and applications literature, by contrast, was drawn primarily from 2000 onward, with particular attention given to the 2020–2026 period. This later window is where graph neural network and quantum-inspired approaches to coloring have begun to appear, and it is also the stretch of literature that most existing surveys have yet to cover.

The inclusion criterion was straightforward: a paper made the cut if it presented a coloring method, an algorithm or a genuine real-world application of coloring, and was left out if coloring only came up in passing. Nor does this review claim to cover every coloring variant that exists

— fractional and circular coloring, for instance, get only a brief mention here rather than sus-tained treatment. The aim throughout was solid, representative coverage of the main coloring types, algorithms and applications, not an exhaustive catalogue of everything that has ever been published on the subject.

3. Fundamental Concepts

3.1 Graphs and Basic Definitions

Graph coloring involves several key ideas and types.

Graph. A graph G is defined as an ordered pair $G = (V(G), E(G))$ where $V(G)$ is a nonempty set of elements called vertices (or nodes) and $E(G) \subseteq \{(u, v) : u, v \in V(G)\}$ is a set of unordered pairs of vertices called edges. The order of a graph is the number of vertices, $n = |V(G)|$; the size is the number of edges, $m = |E(G)|$. A graph with order n and size m is called an (n, m) graph. If an edge $e = (u, v) \in E(G)$, e joins u and v, which are adjacent vertices; e is incident to both u and v. Two distinct edges are adjacent if they share a common vertex.

Graph Coloring. The systematic process of assigning labels ("colors") to graph elements such that no two adjacent elements share a color. Graph coloring is itself an instance of graph labelling [13].

3.2 Vertex Coloring

Vertex coloring assigns a color to each vertex such that for every edge $(u, v) \in E$, $c(u) \neq c(v)$, using as few colors as possible. The minimum number of colors needed is the chromatic number, $\chi(G)$ [13]. Vertex coloring underlies frequency assignment in cellular networks, register allocation in compilers, task scheduling and map coloring in cartography, and determining $\chi(G)$ is NP-hard [25]. Trees and bipartite graphs have $\chi(G) = 2$, while planar graphs require at most four colors by the Four-Color Theorem [2].

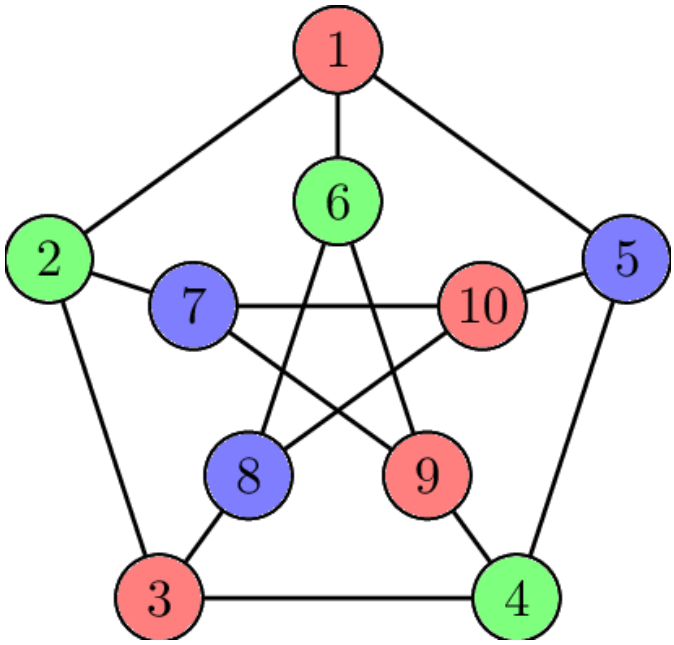

Figure 1: Vertex coloring of the Petersen graph, illustrating a proper 3-coloring.

3.3 Edge Coloring

Edge coloring assigns colors to edges so that no two edges sharing a vertex receive the same color. Formally, an edge coloring is a function $f : E(G) \rightarrow C$ such that for any two edges $e_1, e_2 \in E(G)$ sharing a vertex, $f(e_1) \neq f(e_2)$. The chromatic index $\chi'(G)$ is the minimum number of colors needed for a proper edge coloring [19].

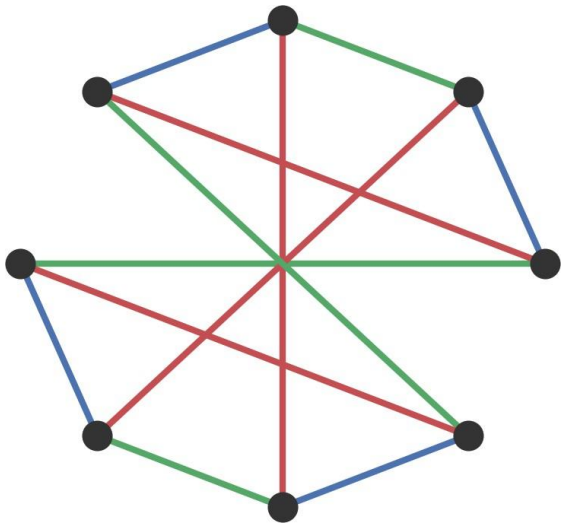

Figure 2: Edge coloring of a prism graph.

3.4 Face Coloring

For a planar graph G with faces $F1, \ldots, Fk$, a face coloring is a mapping $f : F \rightarrow C$ such that any two faces sharing an edge receive different colors. This concept underlies geographic map coloring [2], frequency assignment in wireless communication [22], conflict-free resource assignment in VLSI and network design [40], and register allocation in compiler design [12].

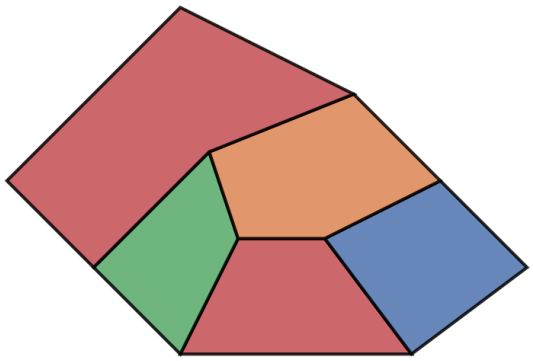

Figure 3: Face coloring of a planar map.

3.5 Total Coloring

Total coloring colors both vertices and edges of $G = (V, E)$ such that no two adjacent vertices, no two adjacent edges and no edge and its incident vertices share a color – unifying vertex and edge coloring into a single framework. Formally, a total coloring is a function $f : V \cup E \to \{1, \ldots, k\}$ such that adjacent or incident elements receive different colors; the total chromatic number $\chi''(G)$ is the smallest such k. Total coloring appears in school and sports-tournament timetabling, frequency assignment in wireless networks, traffic-signal control, register allocation and VLSI design [49].

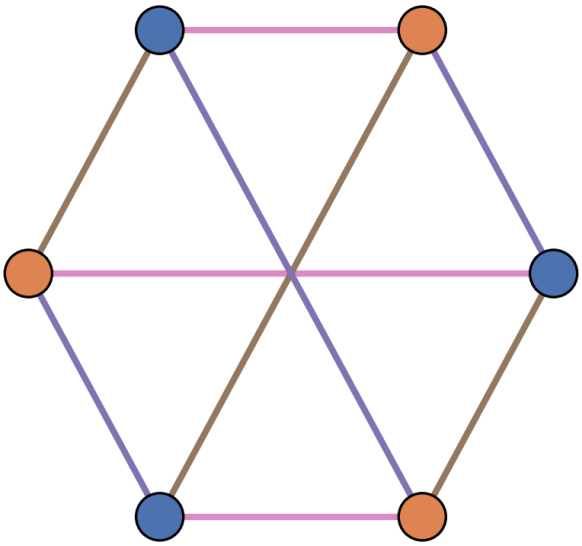

Figure 4: Total coloring of a small graph.

3.6 List Coloring

List coloring, also called choosability, assigns each vertex (or edge) its own list of permitted colors, rather than a single shared palette. Vertex list coloring colors each vertex from its own list so that no two adjacent vertices share a color; edge list coloring does the same for incident edges. This captures scheduling scenarios where local constraints – e.g., a worker's available time slots – vary by element. The concept and its bounds have been studied extensively by Erdős, Rubin and Taylor [18] and by Jensen and Toft [25]. A notable feature of list coloring is that it remains NP-complete even for graph classes, such as interval graphs, where ordinary coloring is solvable in polynomial time [27].

3.7 Graph Classes

Depending on the graph class under study, coloring problems take different forms, since struc-tural restrictions can make coloring easier or harder. Notable classes include:

3.7.1 **Bipartite Graphs:** Graphs were vertices are split into two sets with edges only between sets and are always 2-colorable [49].

3.7.2 **Planar Graphs:** Graphs that are embeddable in a plane without edge crossings and are bounded to four colors by the Four-Color Theorem, and many algorithms exploit this structure [1].

3.7.3 **Interval Graphs:** Graphs whose each vertex corresponds to an interval on the real line, with edges for overlapping intervals. They admit linear-time coloring and appear frequently in scheduling [8].

3.7.4 **Perfect Graphs:** Graphs whose every induced subgraph's chromatic number equals its largest clique size. They generalize several graph families and carry strong theoretical im-plications [15].

3.8 Classical Results and Theorems

Theorem 3.1 (Four Color Theorem)**.** *Every planar graph can be vertex-colored with at most four colors such that no two adjacent vertices receive the same color [2].*

Theorem 3.2 (Brooks' Theorem)**.** *For every connected graph other than a complete graph or an odd cycle,* $\chi(G) \leq \Delta(G)$ *[10].*

Theorem 3.3 (Vizing's Theorem)**.** *For every simple graph G,* $\chi'(G) \in \{\Delta(G), \Delta(G) + 1\}$ *[44].*

4. Algorithms for Graph Coloring

Graph coloring has motivated a substantial body of research spanning theoretical results, heuris-tics and sophisticated metaheuristics. This section surveys the main algorithmic families chrono-logically, highlighting their innovations, methodology and applications, and closes with compar-ative tables that make the trade-offs between families explicit.

4.1 Greedy Algorithm

The greedy algorithm is among the most classical approaches to graph coloring [7]. Given a simple undirected graph $G = (V, E)$, it assigns colors to vertices sequentially so that no two adjacent vertices share a color. For a vertex ordering $(v1, \ldots, vn)$, at step i the algorithm defines

$$A(vi) = \{c(u) : u \in N(vi)\},$$

where $N(vi)$ is the neighborhood of vi, and assigns

$$c(vi) = \min\{k \in \mathrm{N} : k \in / A(vi)\}.$$

This guarantees a proper coloring, though not necessarily an optimal one [7, 23], and the number of colors used depends strongly on vertex ordering. In the worst case the greedy algorithm requires up to $\Delta(G)+1$ colors, where $\Delta(G)$ is the maximum degree [48], but it runs in $O(|V|+|E|)$ time with adjacency lists [7]. Greedy coloring is used extensively in register allocation [11] and scheduling [29], and has motivated refinements such as Welsh-Powell [47] and DSATUR [9].

4.2 Welsh–Powell Algorithm

Since greedy coloring's performance depends heavily on vertex ordering, Welsh and Powell [47] proposed sorting vertices in non-increasing degree order before applying the greedy rule. Pro-cessing highly connected vertices earlier frequently reduces the total number of colors compared with an arbitrary ordering. The method has been applied in timetabling [4], scheduling [27] and cartographic map coloring [50].

4.3 DSATUR Algorithm

Brélaz's DSATUR (Degree of Saturation) algorithm [9] improves on greedy coloring by dynam-ically selecting the next vertex based on its saturation degree,

$\text{sat}(v) = |\{c(u) : u \in N(v),\ c(u) \text{ assigned}\}|,$

coloring the most saturated vertex at each step (ties broken by degree) using the same greedy color-assignment rule. This adaptive choice minimizes redundant color use, making DSATUR especially effective on dense graphs with large chromatic number; it has been applied to register allocation and frequency assignment [11, 27].

4.4 Backtracking

Unlike greedy heuristics, backtracking explores colorings systematically to find an exact solution [5]. For $G = (V, E)$ with color set $\{1, \ldots, k\}$, the goal is a function $c : V \to \{1, \ldots, k\}$ such that $c(u) \neq c(v)$ for every edge $(u, v) \in E$. A vertex vi is assigned a color only if that color does not appear among already-colored neighbors; if no valid color exists, the algorithm backtracks and tries a different assignment. Backtracking is optimal but has worst-case exponential time complexity $O(kn)$, making it impractical for large graphs, though it remains valuable for small or constrained instances and as the basis for exact algorithms [7, 23].

4.5 Metaheuristic Approaches

Given the NP-hardness of graph coloring, exact algorithms such as backtracking are impractical at scale, motivating metaheuristic methods that trade guaranteed optimality for near-optimal solutions in reasonable time via probabilistic and adaptive search [3, 23].

Genetic Algorithm (GA). Population-based, inspired by natural evolution [17]. A candidate coloring is encoded as a chromosome $X = (c(v1), \ldots, c(vn))$, $c(vi) \in \{1, \ldots, k\}$, and a fitness function counts conflicts, $f(X) = |\{(u, v) \in E : c(u) = c(v)\}|$. Selection, crossover and mutation evolve the population toward fewer conflicts. GA-based coloring has succeeded in large-scale scheduling and timetabling [23, 29].

Simulated Annealing (SA). Starts from an initial coloring and iteratively recolors vertices, accepting a new solution with probability $P(\Delta f) = \exp(-\Delta f/T)$, where Δf is the increase in conflicts and T is a temperature parameter that decreases over time, shifting the search from exploration to exploitation. SA has been applied to scheduling and resource allocation [7, 26].

Tabu Search (TS). Minimizes $f(c) = |\{(u, v) \in E : c(u) = c(v)\}|$ by exploring the neigh-borhood of the current solution while maintaining a tabu list of recently visited moves to avoid cycling [24]. Unlike GA's population-based evolution, TS performs guided local search from a single solution with adaptive memory, and has been applied in crew scheduling, examination timetabling and frequency assignment [23, 27].

Ant Colony Optimization (ACO). Artificial ants construct colorings incrementally; the probability of assigning color k to vertex v is proportional to $\tau(v, k)\alpha \cdot \eta(v, k)\beta$, where τ is pheromone intensity and η is heuristic desirability. Pheromone trails reinforce successful as-signments over iterations, converging on high-quality solutions; ACO has been applied in carto-graphic coloring and frequency allocation [7, 23].

Artificial Bee Colony (ABC). Mimics the foraging behavior of honeybee colonies, using employed, onlooker and scout bee roles to balance exploitation of promising colorings with ex-ploration of new regions of the search space [43]. ABC has been explored as an alternative to GA and ACO for coloring problems where a simpler control-parameter set is preferred.

Although GA, SA, TS, ACO and ABC offer no optimality guarantee, they provide scalable, flexible alternatives to exact methods for large and complex graphs where deterministic algo-rithms are impractical, each striking a different balance between exploration and exploitation.

4.6 Emerging Learning-Based and Quantum-Inspired Approaches

A more recent strand of work applies graph neural networks (GNNs) and quantum-inspired optimization to graph coloring, moving beyond the classical and metaheuristic families above. Schuetz et al. [39] introduced a physics inspired GNN (PI-GNN) that optimizes a Potts-model energy formulation of the coloring problem, an approach subsequently extended by Wang et al.

[46] with a negative-message-passing scheme and an entropy term to improve convergence and solution uniformity and by Zhang et al. [52] with a graph isomorphism network whose output is refined with TabuCol. Li et al. [31] examine why standard aggregation-combine GNNs under-perform on coloring specifically, tracing the issue to the heterophilous nature of the problem – adjacent nodes must be assigned different labels – which conflicts with the homophily assump-tion built into most GNN architectures, and propose adjustments accordingly. A related physics-inspired direction avoids neural networks altogether: Bihani and Žerovnik [6] propose a heuristic built on the Ising model, closely related to Boltzmann machines, which adaptively adjusts the number of colors used based on solution quality and is evaluated across the DIMACS benchmark suite with several initialization strategies. In parallel, quantum and quantum-inspired methods have been applied to graph coloring: Titiloye and Crispin [42] showed that quantum anneal-ing can outperform classical simulated annealing on several DIMACS benchmark instances and, Kwok and Pudenz [28] evaluated coloring approximation on quantum annealing hardware. These approaches remain largely experimental and are not yet competitive with mature metaheuristics on most practical instance sizes, but they represent an active and fast-growing research direction that existing graph-coloring surveys predating 2022 do not cover.

4.7 Comparative Summary

Table 1 compares the classical algorithmic families on complexity, solution quality and typical use; Table 2 does the same for the metaheuristic and emerging families.

Table 1: Comparison of classical graph coloring algorithms.

Algorithm	**Time Complexity**	**Solution Quality**	**Best Suited For**	**Ref.**
Greedy	$O(\|V\| + \|E\|)$	Up to $\Delta(G) + 1$ colors	Fast baseline; register allocation	[7]
Welsh-Powell	$O(\|V\|2)$	Better than arbitrary order	Timetabling, map coloring	[47]
DSATUR	$O(\|V\|2)$ typical	Strong on dense graphs	Register allocation, freq. assignment	[9]
Backtracking	$O(kn)$, exponential	Optimal (exact)	Small/constrained instances	[5]

Table 2: Comparison of metaheuristic and emerging graph coloring approaches.

Algorithm	**Mechanism**	**Strength**	**Typical Applications**	**Ref.**
Genetic Algorithm	Population evolution	Highly constrained, large-scale	Scheduling, timetabling	[17]
Simulated Annealing	Probabilistic acceptance	Escapes local minima early	Scheduling, resource alloc.	[26]
Tabu Search	Local search + memory	Avoids cycling	Crew scheduling, exam timetabling	[24]
Ant Colony Opt.	Pheromone reinforcement	Global-local balance	Cartographic coloring	[14]

Artificial Bee Colony	Foraging-inspired search	Fewer control parameters than GA	General-purpose coloring	[43]
GNN-based (PI-GNN, variants)	Learned embeddings, Potts energy	Scales to very large graphs	Large-scale coloring	[39, 46]
Quantum-inspired	QAOA / annealing	Explores via quantum fluctuations	Experimental benchmarks	[42, 28]

5. Applications of Graph Coloring

5.1 Scheduling Problems

Scheduling problems are typically modeled as graph coloring by representing each task as a vertex and drawing an edge between two tasks whenever they cannot share a resource or time slot. A proper coloring of this conflict graph corresponds to a feasible schedule where no two adjacent tasks share a slot: formally, for conflict graph $G = (V, E)$ with $V = \{v1, \ldots, vn\}$ the tasks and E the conflicts, a coloring $f : V \to C$ with $f(u) \neq f(v)$ for every $(u, v) \in E$ and $C = \{1, \ldots, k\}$ the available slots. The minimum such k is $\chi(G)$, the fewest slots needed for a conflict-free schedule.

In exam scheduling, the conflict graph connects two exams sharing common students; a feasible schedule corresponds to a proper coloring, and the chromatic number gives the minimum number of time slots required. Heuristics such as Recursive Largest First (RLF, $O(n2)$) perform well on sparse graphs, while DSATUR performs well on dense instances. List coloring extends this by assigning each vertex its own set of allowable slots, of particular interest in multiprocessor scheduling and aircraft crew assignment.

These ideas find application across a range of real scheduling problems like workforce schedul-ing, airline crew management, nurse rostering and military deployment planning. In airline crew scheduling specifically, tabu search models have been used to minimize backtracking and main-tain feasibility under preferential bidding systems [21]. Nurse scheduling follows a related logic, treating nurses as vertices and shift conflicts as edges, with list coloring producing allocations that are both feasible and more satisfying for staff [37]. Military deployment scheduling instead draws on interval graph coloring, where simple first-fit heuristics have been shown to outper-form integer programming on large-scale unit assignment [51]. Beyond scheduling in the strict sense, coloring-based resource allocation has also reduced execution times in cloud computing by avoiding overlapping tasks [16].

5.2 Cartography

Cartography is one of the most classical applications of graph coloring: maps are represented as planar graphs with vertices as regions and edges as adjacency, and a proper coloring ensures no two adjacent regions share a color while using the fewest colors. The Four-Color Theorem guarantees four colors suffice for any planar map [2]. Formally, for planar graph $G = (V, E)$ with face set $F(G)$, a face-coloring function $f : F(G) \to C$ with $f(F_i) \neq f(F_j)$ whenever $F_i \cap F_j \neq \emptyset$ and $C = \{1, 2, 3, 4\}$. Thomassen's result that planar graphs are five-choosable extended these insights to list coloring [41].

Computational methods have advanced cartographic coloring on several fronts. Hybrid heuristics combining Ant Colony Optimization with greedy coloring have been used to solve four-coloring problems on real-world maps efficiently [14], while more classical tools have also proven durable: Welsh-Powell and greedy algorithms have both been applied to regional maps of Indonesia and West Java, illustrating that older heuristics retain genuine practical value in GIS applications [50, 40]. These uses extend beyond visual clarity, with map coloring now supporting urban planning, political boundary management and GIS-based zoning. Scaling to dynamic or

very large maps remains a genuine challenge, though recent work suggests hybrid metaheuristic approaches may offer a partial solution [14, 41].

5.3 Networks

Graph coloring optimizes network performance in wavelength assignment, frequency assignment and interference reduction. Channels, links or transmitters are graph vertices; edges represent interference constraints; a proper coloring allocates resources so that no two conflicting entities share a resource. Formally, for conflict graph $G = (V, E)$, a coloring $f: V \rightarrow C$ with $f(u) \neq f(v)$ for every $(u, v) \in E$, where C is the set of available frequencies or wavelengths and the minimum $|C|$ is $\chi(G)$ [32].

In optical networks, the Routing and Wavelength Assignment (RWA) problem has been modeled via the Partition Coloring Problem, which partitions vertices into wavelength groups to minimize total wavelengths used [32] and combining partition coloring with routing strategies has reduced blocking probability and improved throughput in WDM networks [35]. Wireless networks use graph coloring for distributed channel allocation without centralized control, im-proving spectrum efficiency while minimizing co-channel interference [33, 36]; similar dynamic coloring algorithms support spectrum optimization in cognitive satellite networks [45]. Recent work applies metaheuristics and learning-based methods to dynamic optical-network congestion reduction [20] and Wi-Fi channel selection [36], reinforcing coloring's role in translating abstract combinatorial optimization into practical network engineering.

5.4 Comparative Summary of Applications

Table 3 maps each application domain to the coloring variant and algorithm family the literature typically applies there.

Table 3: Application domains mapped to their typical coloring variant and algorithm family.

Domain	**Coloring Variant**	**Typical Algorithm(s)**	**Ref.**
Exam / class scheduling	Vertex, list coloring	DSATUR, RLF, GA	[4, 34]
Airline crew / nurse rostering	List coloring	Tabu Search, list heuristics	[21, 37]
Military deployment scheduling	Interval graph coloring	First-fit heuristics	[51]
Cloud task scheduling	Vertex coloring	Chromatic-number allocation	[16]
Cartographic map coloring	Face coloring	Welsh-Powell, greedy, ACO hybrids	[2, 14, 50]

Table 3 – continued from previous page

Domain	Coloring Variant	Typical Algorithm(s)	Ref.
Frequency assignment (cellular/WLAN)	Vertex/face coloring	Distributed coloring heuristics	[22, 38]
Optical network wavelength assignment	Partition coloring	RWA + coloring formulations	[32, 35]
Register allocation (compilers)	Vertex coloring	Greedy, DSATUR	[11, 12]

6. Research Gaps and Future Directions

Despite real progress, a number of open problems and gaps remain in both the theory and the applied algorithmics of graph coloring.

- **Scalability of exact methods.** Computing $\chi(G)$ and related parameters is still NP-hard, which means exact methods like backtracking simply don't scale to the kind of graphs that show up in practice — a continent-scale telecommunications network, for instance. Problem-specific bounds and decomposition strategies seem like the most promising way to close that gap.
- **Open theoretical conjectures.** Some long-standing questions remain genuinely unresolved. Hadwiger's conjecture, linking chromatic number to clique minors, is the best-known example still waiting on a general proof.
- **Dynamic and online coloring.** A great deal of real coloring happens on graphs that don't hold still — traffic networks, wireless spectrum allocation, cloud task scheduling all shift over time — and that calls for algorithms that can update a coloring incrementally rather than starting over from scratch each time something changes.
- **Generalized coloring models.** List coloring and its further generalization, DP-coloring (or correspondence coloring), do a better job than classical coloring at capturing resource-constrained scheduling as it actually plays out — but the algorithmic and complexity theory around these richer models hasn't caught up to the classical case yet.
- **Learning-based methods at scale.** GNN-based coloring approaches (Section 4.6) look promising so far, but mostly on benchmark instances and moderately sized graphs. Whether they can hold their own against mature metaheuristics on the kind of very large, irregular graphs that show up in the real world is still an open question.
- **Quantum approaches beyond proof-of-concept.** Quantum annealing and QAOA-based coloring methods have, so far, only really been tested on small benchmark instances. Getting from there to problem sizes that matter industrially is still an open challenge, and not a purely algorithmic one — the engineering side hasn't caught up either.
- **Hybrid classical-AI pipelines.** Pairing classical metaheuristics with learned heuristics — using the former as a refinement or fallback step, as in [52] — looks like a promising direction. What's missing is a systematic comparison across problem sizes and graph classes to see how well that pairing actually holds up.

Addressing these gaps will likely require combining strengths across the families surveyed in this review – using classical algorithm's guarantees and interpretability alongside metaheuristic's scalability and the pattern generalization capacity of learning-based methods.

7. Conclusion

Graph coloring remains one of the crucial topics in graph theory as it unites deep mathemati-cal foundations with a wide range of real-world applications. This review has worked through the fundamental concepts, classical results and algorithmic techniques that underpin the field, moving from deterministic methods such as Greedy, Welsh-Powell and DSATUR, through meta-heuristics including Genetic Algorithms, Simulated Annealing, Tabu Search and, Ant Colony and Artificial Bee Colony Optimization, to the learning-based and quantum-inspired approaches now emerging. These algorithm families were then mapped onto their principal application do-mains — scheduling, cartography and network optimization — and, departing from prior surveys, compared directly against one another rather than described in sequence, so that the trade-offs between them are made explicit rather than left implicit. Across all of these domains, the chro-matic number and its variants continue to serve as basic measures of conflict resolution and resource allocation.

Considerable challenges remain. Computing chromatic numbers and related parameters is still NP-hard, several long-standing conjectures remain unresolved and dynamic, generalized, and learning-based coloring models are all comparatively underdeveloped next to the static, classical case that dominates the textbook treatment of the subject. Where the field goes from here will likely depend on bringing artificial intelligence, machine learning and quantum computing into closer contact with its classical graph-theoretic foundations, rather than treating them as separate tracks. In that sense, graph coloring continues to do what it has always done well: sit at the productive intersection of theoretical mathematics and practical optimization. Its relevance keeps showing up across mathematics, computer science and engineering alike, which is itself the best argument for continuing to invest in both its theory and its algorithms.

8. Acknowledgments

The authors declare that no funding was received for this research.

REFERENCES

1. Kenneth Appel and Wolfgang Haken. The solution of the four-color-map problem. *Scientific American*, 237(4):108–121, 1977.
2. Kenneth I. Appel and Wolfgang Haken. *Every planar map is four colorable*, volume 98. American Mathematical Soc., 1989.
3. Murat Aslan and Nurdan Akhan Baykan. A performance comparison of graph coloring algorithms. *International Journal of Intelligent Systems and Applications in Engineering*, 4(Special Issue-1):1–7, 2016.
4. Rubul Kumar Bania and Pinkey Duarah. Exam time table scheduling using graph coloring approach. *International Journal of Computer Sciences and Engineering*, 6(5):84–93, 2018.
5. Edward A. Bender and Herbert S. Wilf. A theoretical analysis of backtracking in the graph coloring problem. *Journal of Algorithms*, 6(2):275–282, 1985.
6. Omkar Bihani and Janez Žerovnik. A heuristic for graph coloring based on the ising model. *Mathematics*, 13(18):2976, 2025. doi: 10.3390/math13182976.
7. John Adrian Bondy and Uppaluri Siva Ramachandra Murty. *Graph theory with applications*, volume 290. Macmillan London, 1976.
8. Kellogg S. Booth and George S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using pq-tree algorithms. *Journal of Computer and System Sciences*, 13(3):335–379, 1976.
9. Daniel Brélaz. New methods to color the vertices of a graph. *Communications of the ACM*, 22(4):251–256, 1979.
10. Rowland Leonard Brooks. On colouring the nodes of a network. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 37, pages 194–197. Cambridge University Press, 1941.
11. Gregory J. Chaitin. Register allocation & spilling via graph coloring. *ACM SIGPLAN Notices*, 17(6):98–101, 1982.
12. Gregory J. Chaitin, Marc A. Auslander, Ashok K. Chandra, John Cocke, Martin E. Hopkins, and Peter W. Markstein. Register allocation via coloring. *Computer Languages*, 6(1):47–57, 1981.
13. Gary Chartrand and Ping Zhang. *Chromatic Graph Theory*. Chapman and Hall/CRC, Boca Raton, FL, 2019.
14. Hongshun Chen and Peng Zhou. An ant algorithm for solving the four-coloring map prob-lem. In *2013 Ninth International Conference on Natural Computation (ICNC)*, pages 491– 495. IEEE, 2013.
15. Maria Chudnovsky, Neil Robertson, Paul Seymour, and Robin Thomas. The strong perfect graph theorem. *Annals of Mathematics*, pages 51–229, 2006.
16. Suman De. An efficient technique of resource scheduling in cloud using graph coloring algorithm. *Global Transitions Proceedings*, 3(1):169–176, 2022.

17. Sidi Mohamed Douiri and Souad Elbernoussi. Solving the graph coloring problem via hybrid genetic algorithms. *Journal of King Saud University – Engineering Sciences*, 27(1):114–118, 2015.
18. Paul Erdős, Arthur L. Rubin, and Herbert Taylor. Choosability in graphs. In *Proceedings of the West Coast Conference on Combinatorics, Graph Theory and Computing*, volume 26 of *Congressus Numerantium*, pages 125–157, 1980.
19. Stanley Fiorini and Robin J. Wilson. *Edge-colourings of graphs*. Pitman, London, 1977.
20. Pedro Fonseca, Luís Cancela, and João Rebola. Performance analysis of a graph coloring algorithm for wavelength assignment in dynamic optical networks. In *2022 13th Inter-national Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP)*, pages 534–539. IEEE, 2022.
21. Michel Gamache, Alain Hertz, and Jérôme Olivier Ouellet. A graph coloring model for a feasibility problem in monthly crew scheduling with preferential bidding. *Computers & Operations Research*, 34(8):2384–2395, 2007.
22. Andreas Gamst. Some lower bounds for a class of frequency assignment problems. *IEEE Transactions on Vehicular Technology*, 35(1):8–14, 1986.
23. Martin Charles Golumbic. *Algorithmic Graph Theory and Perfect Graphs*, volume 57 of *Annals of Discrete Mathematics*. Elsevier, Amsterdam, 2nd edition, 2004.
24. Alain Hertz and D. de Werra. Using tabu search techniques for graph coloring. *Computing*, 39(4):345–351, 1987.
25. Tommy R. Jensen and Bjarne Toft. *Graph Coloring Problems*. John Wiley & Sons, Hoboken, NJ, 2011.
26. S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.
27. Marek Kubale. *Graph colorings*, volume 352. American Mathematical Soc., 2004.
28. Julia Kwok and Kristen Pudenz. Graph coloring with quantum annealing. *arXiv preprint arXiv:2012.04470*, 2020.
29. Frank Thomson Leighton. A graph coloring algorithm for large scheduling problems. *Journal of Research of the National Bureau of Standards*, 84(6):489, 1979.
30. Rhyd Lewis. *Guide to Graph Colouring: Algorithms and Applications*. Springer Interna-tional Publishing, Cham, 2nd edition, 2021.
31. Wei Li, Ruofan Li, Yuhan Ma, Siu On Chan, and Bei Yu. Rethinking graph neural networks for the graph coloring problem. *arXiv preprint arXiv:2208.06975*, 2022.
32. Vinod Maan and G. N. Purohit. A distributed approach for frequency allocation using graph coloring in mobile networks. *International Journal of Computer Applications*, 58(6), 2012.
33. Jacob A. Mack. *An Application of Graph Coloring to a Scheduling Problem*. PhD thesis, Naval Postgraduate School, Monterey, California, 1968.
34. Dániel Marx. Graph colouring problems and their applications in scheduling. *Periodica Polytechnica Electrical Engineering (Archives)*, 48(1–2):11–16, 2004.
35. Thiago F. Noronha and Celso C. Ribeiro. Routing and wavelength assignment by partition colouring. *European Journal of Operational Research*, 171(3):797–810, 2006.
36. David Orden, Jose Manuel Gimenez-Guzman, Ivan Marsa-Maestre, and Enrique De la Hoz. Spectrum graph coloring and applications to wi-fi channel assignment. *Symmetry*, 10(3): 65, 2018.
37. A. Perera. Automated system for nurse scheduling using graph coloring. Mathematics Department, Faculty of Science, Peradeniya University, Sri Lanka, 2011.
38. Janne Riihijärvi, Marina Petrova, and Petri Mähönen. Frequency allocation for wlans using graph colouring techniques. In *Second Annual Conference on Wireless On-demand Network Systems and Services (WONS 2005)*, pages 71–78. IEEE, 2005.
39. M. J. Schuetz, J. K. Brubaker, Z. Zhu, and H. G. Katzgraber. Graph coloring with physics-inspired graph neural networks. *Physical Review Research*, 4(4):043131, 2022.
40. T. N. Sipayung, S. Suwilo, and Parapat Gultom. Implementation of the greedy algorithm on graph coloring. In *Journal of Physics: Conference Series*, volume 2157, page 012003. IOP Publishing, 2022.
41. Carsten Thomassen. Every planar graph is 5-choosable. *Journal of Combinatorial Theory, Series B*, 62(1):180–181, 1994.
42. Olawale Titiloye and Alan Crispin. Quantum annealing of the graph coloring problem. *Discrete Optimization*, 8(2):376–384, 2011.
43. Pei-Wei Tsai et al. Enhanced artificial bee colony optimization. *International Journal of Innovative Computing, Information and Control*, 5(12):5081–5092, 2009.
44. Vadim G. Vizing. On an estimate of the chromatic class of a p-graph. *Diskret Analiz*, 3: 25–30, 1964.
45. Li Wang, Kwok-Yan Lam, Jiangxin Zhang, and Feng Li. Spectrum optimization in cognitive satellite networks with graph coloring method. *Wireless Networks*, pages 1–10, 2019.
46. Xiangyu Wang, Xueming Yan, and Yaochu Jin. A graph neural network with negative message passing and uniformity maximization for graph coloring. *Complex & Intelligent Systems*, 10(3):4445–4455, 2024.
47. Dominic J. A. Welsh and Martin B. Powell. An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal*, 10(1):85–86, 1967.
48. Douglas Brent West et al. *Introduction to Graph Theory*, volume 2. Prentice Hall, Upper Saddle River, 2001.
49. H. P. Yap. *Total Colourings of Graphs*, volume 1623 of *Lecture Notes in Mathematics*. Springer, Berlin, 1996.
50. Dimas Dani Zaini, Howsen Vincensius, Kevin Aurelian Nathanielle Untung Widjaja, Nurhasanah, and Alif Tri Handoyo. Implementing welsh-powell algorithm on coloring the map of west java. In *Proceedings of the 8th International Conference on Sustainable Infor-mation Engineering and Technology*, pages 679–684, 2023.

51. Mark Zais and Manuel Laguna. A graph coloring approach to the deployment scheduling and unit assignment problem. *Journal of Scheduling*, 19:73–90, 2016.
52. Yicheng Zhang, Kai Zhang, and Ni Wu. Using graph isomorphism network to solve graph coloring problem. In *2024 4th International Conference on Consumer Electronics and Com-puter Engineering (ICCECE)*, pages 209–215. IEEE, 2024.