

Online Graph Coloring: Baselines, Degree Heuristics, and Empirical Results

COMP 6651 Project

December 2, 2025

1 Problem Description

Graph coloring represents a fundamental area of study in graph theory, originating from the famous Four Color Problem first posed in 1852 [1]. In online coloring, an algorithm must assign colors to graph vertices as they appear sequentially, without the ability to preview future vertices or modify previously assigned colors [3]. Research has shown that any such online approach may need at minimum $(2n/(\log n)^2)\chi(G)$ colors under worst-case conditions [4].

The FirstFit algorithm stands out as perhaps the most straightforward online graph coloring method [3]. This greedy approach works by establishing a predetermined ordering of available colors, then assigning each new vertex the lowest-ranked color that preserves proper coloring constraints. Researchers have studied this algorithm extensively across various types of graphs [3, 5, 6]. A different notable method, CBIP (Coloring Based on Interval Partitioning), was developed specifically for bipartite graphs: as each vertex $v = \sigma(i)$ is presented, the algorithm identifies the complete connected component containing v within the currently known portion of the graph [7].

While FirstFit and CBIP provide foundational strategies, recent studies have explored enhancements through degree-based heuristics. For instance, reordering vertices based on their degrees before coloring can lead to improved performance [5]. Additionally, batch processing techniques, where vertices are grouped and reordered within batches, have shown promise in reducing the number of colors used [2].

The purpose of this project is to implement, test, and empirically compare simple online baselines and degree-based heuristics under reproducible conditions. We sweep graph size n and density p , and analyze how ordering vertices by degree and per-batch degree reordering affect the efficiency of greedy coloring.

2 Implementation Details

Graph Data Structure

We use a lightweight, mutable, undirected graph. Vertices are positive integers; adjacency maps vertex IDs to sets of neighbors. Self-loops are ignored, and edges are stored symmetrically.

Core Graph Functions

Data structures: adjacency map $\text{adj} : V \rightarrow 2^V$ (sets of neighbours).

INIT_GRAPH(n):

```
define  $\text{adj}(i) \leftarrow \emptyset$  for  $i \in [1..n]$ ;  

return  $\text{adj}$ 
```

ENSURE_VERTEX(adj, v):

```
if  $v \notin \text{adj}$  then  

 $\text{adj}(v) \leftarrow \emptyset$ 
```

ADD_EDGE(adj, u, v):

```
if  $u = v$  then return;  

ENSURE_VERTEX on  $u, v$ ;  

 $\text{adj}(u) \leftarrow \text{adj}(u) \cup \{v\}$ ;  

 $\text{adj}(v) \leftarrow \text{adj}(v) \cup \{u\}$ 
```

NEIGHBOURS(adj, v):

```
return  $\text{adj}(v)$  ▷ empty if unseen
```

DEGREE(adj, v):

```
return  $|\text{adj}(v)|$ 
```

CONNECTED_COMPONENT($\text{adj}, s, \text{Allowed}$):

```
if  $s \notin \text{Allowed}$  then return  $\emptyset$   

 $\mathcal{C} \leftarrow \{s\}$ ;  $Q \leftarrow [s]$   

while  $Q$  not empty do  

 $x \leftarrow \text{pop\_front}(Q)$   

for  $y \in \text{adj}(x)$  do  

if ( $y \in \text{Allowed}$ )  $\wedge (y \notin \mathcal{C})$  then  

 $\mathcal{C} \leftarrow \mathcal{C} \cup \{y\}$ ;  $\text{push\_back}(Q, y)$   

return  $\mathcal{C}$ 
```

BIPARTITION_COMPONENT(adj, \mathcal{C}):

```
color  $\leftarrow \emptyset$ ;  $A \leftarrow \emptyset$ ;  $B \leftarrow \emptyset$   

for each  $v \in \mathcal{C}$  do  

if  $v \in \text{color}$  then continue  

color  $\leftarrow \emptyset$ ;  $A \leftarrow \emptyset$ ;  $B \leftarrow \emptyset$   

 $Q \leftarrow [v]$ ;  $\text{color}(v) \leftarrow 0$ ;  $A \leftarrow A \cup \{v\}$   

while  $Q$  not empty do  

 $x \leftarrow \text{pop\_front}(Q)$   

for  $y \in \text{adj}(x) \cap \mathcal{C}$  do  

if  $y \notin \text{color}$  then  

 $\text{color}(y) \leftarrow 1 - \text{color}(x)$   

 $(A \text{ if } \text{color}(y) = 0 \text{ else } B) \leftarrow \cdot \cup \{y\}$   

 $\text{push\_back}(Q, y)$   

return  $(A, B)$ 
```

Generator

We construct k -colorable graphs reproducibly by partitioning vertices into k independent sets and adding cross edges with probability p .

```

Generate( $n, k, p$ )
// Output: graph  $G = (V, E)$  and online order  $\sigma$ 
 $S[1..k] \leftarrow k$  empty sets                                ▷ partition of  $V$ 
for  $i = 1..k$  do  $S[i] \leftarrow S[i] \cup \{i\}$                       ▷ seed non-empty
for  $v = k + 1..n$  do  $S[\text{rand}(1..k)] \leftarrow S[\cdot] \cup \{v\}$ 
 $G \leftarrow \text{INIT\_GRAPH}(n)$ 
for  $i = 1..k$  do
    for each  $v \in S[i]$  do
        for  $j = 1..k, j \neq i$  do
            choose  $u \in S[j]$  uniformly; ADD_EDGE( $G, v, u$ )      ▷ mandatory cross edge
            for each  $u_2 \in S[j] \setminus \{u\}$  do
                if  $\text{rand}() < p$  then ADD_EDGE( $G, v, u_2$ )      ▷ random cross edges
        define  $\sigma([1..n]) \leftarrow V$  and uniformly randomize order
    return  $(G, \sigma)$                                          ▷ online presentation order

```

Algorithms

We retain four algorithms; pseudocode mirrors the Python implementations.

- **FirstFit (online)**: Greedy assignment of the smallest permissible color by arrival order.
- **CBIP (online, bipartite-specific)**: Uses component bipartition; reported for $k = 2$.
- **HeuristicDegree (semi-online reorder)**: Reorders the arrival sequence by non-increasing degree, then applies FirstFit.
- **BatchDegree (semi-online)**: Processes batches of size t , reorders within batch by degree, colors via FirstFit.

```

FirstFit( $G, \sigma$ )
// Input:  $G = (V, E)$ : Current online graph,  $\sigma(1..|V|)$ : The presentation order of the vertices.
define  $c(V)$                                               ▷ stores coloring of  $V$ 
initialize  $c(V)$  to nil                                    ▷ no colors set yet
 $i \leftarrow 1$ 
define vertex  $v$ , set of vertices  $N$ 
while  $i \leq |V|$  do
     $v \leftarrow \sigma(i)$ 
     $N \leftarrow$  all neighbours of  $v$  in  $V \cap \sigma[1..i - 1]$       ▷ include only current revealed vertices
     $c(v) \leftarrow \min(\mathbb{N} \setminus c(N))$                       ▷  $c(N)$  the set of colors of vertices in  $N$ 
     $i \leftarrow i + 1$ 
return  $c$ 

```

```

HeuristicDegree( $G$ )
// Input:  $G = (V, E)$  — Current offline reorder followed by online greedy
    compute  $d(v)$  for all  $v \in V$                                 ▷ static degree in  $G$ 
    define  $\sigma([1..n]) \leftarrow V$ 
    sort  $\sigma$  by  $d(v)$  in non-increasing order                      ▷ highest degree first
    return FirstFit( $G$ ) on  $\sigma$ 

```

```

CBIP( $G$ )
// Input:  $G = (V, E)$  — Online bipartite coloring (reported for  $k = 2$ )
    initialize  $c(V)$  to nil;  $R \leftarrow \emptyset$                          ▷  $R$ : revealed vertices
    define  $\sigma([1..n]) \leftarrow V$ 
    for  $i = 1$  to  $|V|$  do
         $v \leftarrow \sigma(i)$  ;  $R \leftarrow R \cup \{v\}$ 
         $\mathcal{C} \leftarrow \text{CONNECTEDCOMPONENT}(G, v, R)$ 
         $(A, B) \leftarrow \text{BIPARTITION}(G, \mathcal{C})$ 
        if ( $v \notin A$ ) and ( $v \in B$ ) then swap( $A, B$ )
         $U \leftarrow \{c(u) : u \in (B \cap R)\}$                                ▷ colors used on the opposite side
         $c(v) \leftarrow \min(\mathbb{N} \setminus U)$ 
    return  $c$ 

```

```

BatchDegree( $G, t$ )
// Input:  $G = (V, E)$  — Semi-online batches of size  $t$  with degree reorder
    initialize  $c(V)$  to nil;  $R \leftarrow \emptyset$  ; compute  $d(v)$  for all  $v$ 
    define  $\sigma([1..n]) \leftarrow V$ 
    for  $s \in \{1, 1+t, 1+2t, \dots\}$  do
         $B \leftarrow \sigma[s : s+t]$                                          ▷ current batch
        reorder  $B$  by  $d(v)$  in non-increasing order
        for each  $v \in B$  do
             $N \leftarrow \text{neighbours of } v \text{ in } R$ 
             $c(v) \leftarrow \min(\mathbb{N} \setminus c(N))$ 
             $R \leftarrow R \cup \{v\}$ 
    return  $c$ 

```

3 Implementation correctness

- **Software correctness:** Unit tests validate that self-loops are ignored, parallel edges are not duplicated (set-based adjacency), adjacency is symmetric, and EDGES I/O is consistent (each undirected edge written once, tolerant reader).
- **Algorithm correctness (examples):** FirstFit yields proper colorings on random k -colorable graphs and uses 3 colors on a triangle; CBIP colors bipartite instances properly (e.g., 4-cycle uses at most 2 colors). We manually verified outputs on small graphs with known chromatic numbers, and used boundary testing to ensure robustness.

4 Experimental Setup

Our experimental framework is implemented in Python 3.13.7 and organized as follows:

- **Parameter grid:** Iterate over $n \in \{50, 100, 150, 200, 300, 400, 500, 600, 700, 800, 900, 1000\}$, $k \in \{2, 3, 4\}$, $p \in \{0.05, 0.1, 0.2, 0.3\}$; for each triple generate $N = 100$ independent instances.
- **Deterministic artifacts:** For every instance we persist two plain-text files: an `xxx.edges` file (one undirected edge $u v$ per line, $u < v$) and an `xxx.order` file (vertex arrival permutation σ).
- **Reproducibility:** Algorithms never sample randomness internally when consuming a stored instance; they read the pre-generated edge list and vertex order, ensuring identical outcomes across runs/machines.
- **Graph generation (generator/generate.py):** Implements the partition-based k -colorable construction plus probabilistic cross edges with probability p ; writes artifacts only if absent (idempotent cache behavior).
- **Core structure (graph.py):** Provides adjacency maintenance, component discovery, and bipartition logic used by CBIP; omits multi-edges and self-loops.
- **Algorithms (algorithms/*.py):** Each file exposes a pure function mapping (G, σ) (and batch size where relevant) to a coloring dictionary. No global state, facilitating multiprocessing.
- **Simulation driver (simulations.py):** Orchestrates sweeps over the parameter grid, loads cached instances, dispatches algorithm runs, aggregates per-instance color counts, and computes competitive ratios.
- **Metrics (metrics.py):** Defines helper routines (e.g., competitive ratio, summary statistics: mean, standard deviation) separated for clarity and testability.
- **Parallelism:** Uses Python `multiprocessing` (process pool) for independent instances within the same (n, k, p) slice when algorithms are stateless. `BatchDegree` may be run in the main process to avoid serialization of internal lambdas.
- **Progress aggregation:** Intermediate results accumulated into in-memory structures, then flushed periodically to CSVs under `outputs/` (one per algorithm variant plus a consolidated enriched summary).
- **Failure tolerance:** Instance-level exceptions (should none occur in normal operation) are logged and skipped without aborting the entire sweep.
- **Main entry (main.py):** Provides CLI handling (e.g., selecting algorithms, enabling parallel mode, specifying output paths) and invokes simulation routines; supports partial grids for quick exploratory runs.
- **Test coverage (tests/test_sanity.py):** Validates structural invariants (symmetry, loop suppression) and correctness of edge writing/reading to guarantee reliability of persisted dataset before large simulations run.
- **Post-processing (notebooks / plot_results.ipynb):** Loads summary CSVs, produces fit tables, residual plots, and family figures; writes PNG outputs consumed by LaTeX report.

5 Results

We report two simulation tracks. Each uses fixed instance artifacts (edge list and arrival order) for strict reproducibility and repeats $N = 100$ times per (n, k, p) .

5.1 Simulation I: FirstFit and CBIP

Parameters and setup

- **Order:** Use the persisted online order σ per instance; no reshuffling at evaluation time.
- **FirstFit:** No tunable parameter; assigns smallest feasible colour on arrival.
- **CBIP:** Reported only for $k = 2$ (bipartite inputs). Computes the revealed component and a bipartition on-the-fly; colours from the opposite side's palette.
- **Repetitions:** $N = 100$ per (n, k, p) ; performance summarised by mean and standard deviation of the competitive ratio.

Table 1: FirstFit: results across $k \in \{2, 3, 4\}$ with density p .

Algorithm	k	p	n	N	$\rho(\text{Alg})$	$SD(\rho(\text{Alg}))$
FirstFit	2	0.05	50	100	2.04	0.31
FirstFit	2	0.05	100	100	2.52	0.43
FirstFit	2	0.05	150	100	2.8	0.53
FirstFit	2	0.05	200	100	3.25	0.54
FirstFit	2	0.05	300	100	3.7	0.77
FirstFit	2	0.05	400	100	4.16	0.69
FirstFit	2	0.05	500	100	4.26	0.95
FirstFit	2	0.05	600	100	4.58	0.93
FirstFit	2	0.05	700	100	4.7	1.15
FirstFit	2	0.05	800	100	4.96	1.22
FirstFit	2	0.05	900	100	5.29	1.23
FirstFit	2	0.05	1000	100	5.15	1.23
FirstFit	3	0.05	50	100	1.87	0.19
FirstFit	3	0.05	100	100	2.36	0.19
FirstFit	3	0.05	150	100	2.73	0.24
FirstFit	3	0.05	200	100	3.1	0.2
FirstFit	3	0.05	300	100	3.78	0.23
FirstFit	3	0.05	400	100	4.31	0.25
FirstFit	3	0.05	500	100	4.82	0.36
FirstFit	3	0.05	600	100	5.37	0.31
FirstFit	3	0.05	700	100	5.72	0.4
FirstFit	3	0.05	800	100	6.22	0.4
FirstFit	3	0.05	900	100	6.61	0.43
FirstFit	3	0.05	1000	100	6.94	0.54
FirstFit	4	0.05	50	100	1.63	0.15

Continued on next page

Algorithm	k	p	n	N	$\overline{\rho(Alg)}$	$SD(\rho(Alg))$
FirstFit	4	0.05	100	100	2.04	0.14
FirstFit	4	0.05	150	100	2.37	0.13
FirstFit	4	0.05	200	100	2.68	0.16
FirstFit	4	0.05	300	100	3.23	0.15
FirstFit	4	0.05	400	100	3.79	0.15
FirstFit	4	0.05	500	100	4.24	0.17
FirstFit	4	0.05	600	100	4.71	0.18
FirstFit	4	0.05	700	100	5.12	0.17
FirstFit	4	0.05	800	100	5.57	0.16
FirstFit	4	0.05	900	100	5.95	0.19
FirstFit	4	0.05	1000	100	6.36	0.2
FirstFit	2	0.1	50	100	2.11	0.55
FirstFit	2	0.1	100	100	2.51	0.8
FirstFit	2	0.1	150	100	2.91	0.96
FirstFit	2	0.1	200	100	2.89	1.05
FirstFit	2	0.1	300	100	3.14	1.11
FirstFit	2	0.1	400	100	3.37	1.41
FirstFit	2	0.1	500	100	3.66	1.55
FirstFit	2	0.1	600	100	3.57	1.54
FirstFit	2	0.1	700	100	3.37	1.68
FirstFit	2	0.1	800	100	3.88	1.8
FirstFit	2	0.1	900	100	4.02	1.87
FirstFit	2	0.1	1000	100	3.6	1.71
FirstFit	3	0.1	50	100	2.08	0.29
FirstFit	3	0.1	100	100	2.86	0.29
FirstFit	3	0.1	150	100	3.22	0.45
FirstFit	3	0.1	200	100	3.77	0.48
FirstFit	3	0.1	300	100	4.54	0.65
FirstFit	3	0.1	400	100	5.01	0.79
FirstFit	3	0.1	500	100	5.67	0.89
FirstFit	3	0.1	600	100	5.9	1.01
FirstFit	3	0.1	700	100	6.58	1.13
FirstFit	3	0.1	800	100	6.61	1.28
FirstFit	3	0.1	900	100	7.15	1.59
FirstFit	3	0.1	1000	100	7.43	1.39
FirstFit	4	0.1	50	100	1.88	0.15
FirstFit	4	0.1	100	100	2.47	0.19
FirstFit	4	0.1	150	100	3.03	0.21
FirstFit	4	0.1	200	100	3.52	0.24
FirstFit	4	0.1	300	100	4.35	0.32
FirstFit	4	0.1	400	100	5.13	0.32
FirstFit	4	0.1	500	100	5.78	0.42
FirstFit	4	0.1	600	100	6.43	0.36
FirstFit	4	0.1	700	100	6.98	0.62
FirstFit	4	0.1	800	100	7.5	0.52

Continued on next page

Algorithm	k	p	n	N	$\overline{\rho(Alg)}$	$SD(\rho(Alg))$
FirstFit	4	0.1	900	100	7.95	0.7
FirstFit	4	0.1	1000	100	8.47	0.8
FirstFit	2	0.2	50	100	1.72	0.7
FirstFit	2	0.2	100	100	1.98	0.86
FirstFit	2	0.2	150	100	1.95	0.83
FirstFit	2	0.2	200	100	2.12	1.1
FirstFit	2	0.2	300	100	2	0.99
FirstFit	2	0.2	400	100	1.97	1
FirstFit	2	0.2	500	100	2.14	1.08
FirstFit	2	0.2	600	100	2.12	1.07
FirstFit	2	0.2	700	100	2.19	1.06
FirstFit	2	0.2	800	100	2.23	1.26
FirstFit	2	0.2	900	100	2.05	1.19
FirstFit	2	0.2	1000	100	1.97	1.02
FirstFit	3	0.2	50	100	2.16	0.52
FirstFit	3	0.2	100	100	2.69	0.81
FirstFit	3	0.2	150	100	2.98	1.07
FirstFit	3	0.2	200	100	3.15	1.24
FirstFit	3	0.2	300	100	3.45	1.25
FirstFit	3	0.2	400	100	3.73	1.6
FirstFit	3	0.2	500	100	3.87	1.89
FirstFit	3	0.2	600	100	3.97	1.88
FirstFit	3	0.2	700	100	3.72	1.97
FirstFit	3	0.2	800	100	3.73	1.76
FirstFit	3	0.2	900	100	4.02	1.9
FirstFit	3	0.2	1000	100	3.69	1.9
FirstFit	4	0.2	50	100	2.17	0.32
FirstFit	4	0.2	100	100	2.71	0.53
FirstFit	4	0.2	150	100	3.38	0.67
FirstFit	4	0.2	200	100	3.73	1.02
FirstFit	4	0.2	300	100	4.59	1.14
FirstFit	4	0.2	400	100	4.76	1.47
FirstFit	4	0.2	500	100	5.46	1.62
FirstFit	4	0.2	600	100	5.88	1.76
FirstFit	4	0.2	700	100	5.89	1.8
FirstFit	4	0.2	800	100	5.86	1.93
FirstFit	4	0.2	900	100	5.83	2.5
FirstFit	4	0.2	1000	100	5.9	2.25
FirstFit	2	0.3	50	100	1.43	0.49
FirstFit	2	0.3	100	100	1.61	0.75
FirstFit	2	0.3	150	100	1.46	0.57
FirstFit	2	0.3	200	100	1.44	0.59
FirstFit	2	0.3	300	100	1.49	0.57
FirstFit	2	0.3	400	100	1.6	0.68
FirstFit	2	0.3	500	100	1.47	0.58

Continued on next page

Algorithm	k	p	n	N	$\overline{\rho(Alg)}$	$SD(\rho(Alg))$
FirstFit	2	0.3	600	100	1.47	0.49
FirstFit	2	0.3	700	100	1.52	0.55
FirstFit	2	0.3	800	100	1.53	0.68
FirstFit	2	0.3	900	100	1.51	0.64
FirstFit	2	0.3	1000	100	1.64	0.74
FirstFit	3	0.3	50	100	1.81	0.63
FirstFit	3	0.3	100	100	2.07	0.94
FirstFit	3	0.3	150	100	2.08	1.06
FirstFit	3	0.3	200	100	2.21	1.07
FirstFit	3	0.3	300	100	2.17	0.98
FirstFit	3	0.3	400	100	2.26	1.05
FirstFit	3	0.3	500	100	2.32	1.09
FirstFit	3	0.3	600	100	2.06	0.97
FirstFit	3	0.3	700	100	2.17	1.02
FirstFit	3	0.3	800	100	2.15	1.02
FirstFit	3	0.3	900	100	2.2	0.97
FirstFit	3	0.3	1000	100	2.35	1.13
FirstFit	4	0.3	50	100	2.01	0.45
FirstFit	4	0.3	100	100	2.4	0.75
FirstFit	4	0.3	150	100	2.56	0.98
FirstFit	4	0.3	200	100	2.82	1.13
FirstFit	4	0.3	300	100	2.8	1.1
FirstFit	4	0.3	400	100	3.05	1.24
FirstFit	4	0.3	500	100	2.8	1.19
FirstFit	4	0.3	600	100	2.96	1.26
FirstFit	4	0.3	700	100	3.11	1.64
FirstFit	4	0.3	800	100	3.02	1.27
FirstFit	4	0.3	900	100	2.98	1.28
FirstFit	4	0.3	1000	100	3.08	1.47

Table 2: CBIP: results for bipartite inputs ($k = 2$) with density p .

Algorithm	k	p	n	N	$\overline{\rho(Alg)}$	$SD(\rho(Alg))$
CBIP	2	0.05	50	100	1.91	0.26
CBIP	2	0.05	100	100	1.99	0.27
CBIP	2	0.05	150	100	2.01	0.27
CBIP	2	0.05	200	100	2.05	0.21
CBIP	2	0.05	300	100	2.06	0.28
CBIP	2	0.05	400	100	2.05	0.23
CBIP	2	0.05	500	100	2.04	0.27
CBIP	2	0.05	600	100	2.05	0.25
CBIP	2	0.05	700	100	2.07	0.22

Continued on next page

Algorithm	k	p	n	N	$\overline{\rho(Alg)}$	$SD(\rho(Alg))$
CBIP	2	0.05	800	100	2.07	0.25
CBIP	2	0.05	900	100	2.06	0.25
CBIP	2	0.05	1000	100	2.05	0.21
CBIP	2	0.1	50	100	1.76	0.33
CBIP	2	0.1	100	100	1.78	0.32
CBIP	2	0.1	150	100	1.85	0.29
CBIP	2	0.1	200	100	1.79	0.3
CBIP	2	0.1	300	100	1.88	0.29
CBIP	2	0.1	400	100	1.87	0.35
CBIP	2	0.1	500	100	1.86	0.3
CBIP	2	0.1	600	100	1.87	0.28
CBIP	2	0.1	700	100	1.83	0.31
CBIP	2	0.1	800	100	1.88	0.32
CBIP	2	0.1	900	100	1.84	0.3
CBIP	2	0.1	1000	100	1.86	0.29
CBIP	2	0.2	50	100	1.47	0.38
CBIP	2	0.2	100	100	1.56	0.38
CBIP	2	0.2	150	100	1.56	0.36
CBIP	2	0.2	200	100	1.56	0.39
CBIP	2	0.2	300	100	1.55	0.39
CBIP	2	0.2	400	100	1.55	0.37
CBIP	2	0.2	500	100	1.59	0.35
CBIP	2	0.2	600	100	1.57	0.35
CBIP	2	0.2	700	100	1.64	0.35
CBIP	2	0.2	800	100	1.61	0.34
CBIP	2	0.2	900	100	1.54	0.36
CBIP	2	0.2	1000	100	1.55	0.37
CBIP	2	0.3	50	100	1.35	0.36
CBIP	2	0.3	100	100	1.4	0.41
CBIP	2	0.3	150	100	1.33	0.35
CBIP	2	0.3	200	100	1.3	0.33
CBIP	2	0.3	300	100	1.38	0.32
CBIP	2	0.3	400	100	1.4	0.34
CBIP	2	0.3	500	100	1.33	0.37
CBIP	2	0.3	600	100	1.36	0.33
CBIP	2	0.3	700	100	1.41	0.38
CBIP	2	0.3	800	100	1.32	0.34
CBIP	2	0.3	900	100	1.34	0.34
CBIP	2	0.3	1000	100	1.41	0.4

5.2 Simulation II: HeuristicDegree and BatchDegree

Parameterisation and heuristic choices

- **HeuristicDegree:** Offline reorder by non-increasing static degree, then FirstFit. No extra

parameter beyond using the same fixed σ as baseline to define V .

- **BatchDegree:** Semi-online with batch size t ; within each batch, reorder by static degree and colour via FirstFit. We report $t \in \{10, 50\}$ as representative settings, chosen for a good speed/quality trade-off observed in pilot runs.
- **Alternatives explored (not selected):** We evaluated several on-arrival colouring heuristics: (i) *Randomised colour* tie-breaking, (ii) *MRU* preference (biasing towards most recently used colour), and (iii) *Least-used* global colour selection. Across (n, p) , these yielded competitive ratios that were comparable to or worse than FirstFit and significantly worse than BatchDegree, especially at higher densities. Consequently, we focus on BatchDegree for clearer improvements.
- **Impact of t :** Larger t generally reduces colours by enabling more effective local ordering, with diminishing returns beyond $t \approx 50$ at our scales.

Table 3: HeuristicDegree: degree-sorted FirstFit across $k \in \{2, 3, 4\}$ with density p .

Algorithm	k	p	n	N	$\overline{\rho(Alg)}$	$SD(\rho(Alg))$
FirstFit(HeuristicDegree)	2	0.05	50	100	1.37	0.42
FirstFit(HeuristicDegree)	2	0.05	100	100	1.68	0.64
FirstFit(HeuristicDegree)	2	0.05	150	100	1.96	0.79
FirstFit(HeuristicDegree)	2	0.05	200	100	2.14	0.9
FirstFit(HeuristicDegree)	2	0.05	300	100	2.56	1.03
FirstFit(HeuristicDegree)	2	0.05	400	100	2.7	1.17
FirstFit(HeuristicDegree)	2	0.05	500	100	2.54	1.33
FirstFit(HeuristicDegree)	2	0.05	600	100	2.95	1.44
FirstFit(HeuristicDegree)	2	0.05	700	100	3.02	1.53
FirstFit(HeuristicDegree)	2	0.05	800	100	3.14	1.48
FirstFit(HeuristicDegree)	2	0.05	900	100	3.51	1.59
FirstFit(HeuristicDegree)	2	0.05	1000	100	3.39	1.7
FirstFit(HeuristicDegree)	3	0.05	50	100	1.59	0.19
FirstFit(HeuristicDegree)	3	0.05	100	100	1.99	0.24
FirstFit(HeuristicDegree)	3	0.05	150	100	2.37	0.24
FirstFit(HeuristicDegree)	3	0.05	200	100	2.69	0.36
FirstFit(HeuristicDegree)	3	0.05	300	100	3.21	0.32
FirstFit(HeuristicDegree)	3	0.05	400	100	3.66	0.46
FirstFit(HeuristicDegree)	3	0.05	500	100	4.01	0.51
FirstFit(HeuristicDegree)	3	0.05	600	100	4.54	0.57
FirstFit(HeuristicDegree)	3	0.05	700	100	4.75	0.66
FirstFit(HeuristicDegree)	3	0.05	800	100	5.02	0.73
FirstFit(HeuristicDegree)	3	0.05	900	100	5.35	0.85
FirstFit(HeuristicDegree)	3	0.05	1000	100	5.66	1
FirstFit(HeuristicDegree)	4	0.05	50	100	1.46	0.12
FirstFit(HeuristicDegree)	4	0.05	100	100	1.81	0.12
FirstFit(HeuristicDegree)	4	0.05	150	100	2.11	0.14
FirstFit(HeuristicDegree)	4	0.05	200	100	2.4	0.14

Continued on next page

Algorithm	k	p	n	N	$\overline{\rho(Alg)}$	$SD(\rho(Alg))$
FirstFit(HeuristicDegree)	4	0.05	300	100	2.91	0.14
FirstFit(HeuristicDegree)	4	0.05	400	100	3.34	0.2
FirstFit(HeuristicDegree)	4	0.05	500	100	3.8	0.18
FirstFit(HeuristicDegree)	4	0.05	600	100	4.19	0.21
FirstFit(HeuristicDegree)	4	0.05	700	100	4.65	0.19
FirstFit(HeuristicDegree)	4	0.05	800	100	4.97	0.26
FirstFit(HeuristicDegree)	4	0.05	900	100	5.31	0.23
FirstFit(HeuristicDegree)	4	0.05	1000	100	5.7	0.28
FirstFit(HeuristicDegree)	2	0.1	50	100	1.35	0.51
FirstFit(HeuristicDegree)	2	0.1	100	100	1.44	0.64
FirstFit(HeuristicDegree)	2	0.1	150	100	1.54	0.81
FirstFit(HeuristicDegree)	2	0.1	200	100	1.58	0.93
FirstFit(HeuristicDegree)	2	0.1	300	100	1.68	1.02
FirstFit(HeuristicDegree)	2	0.1	400	100	1.74	1.09
FirstFit(HeuristicDegree)	2	0.1	500	100	1.86	1.27
FirstFit(HeuristicDegree)	2	0.1	600	100	1.82	1.22
FirstFit(HeuristicDegree)	2	0.1	700	100	1.84	1.32
FirstFit(HeuristicDegree)	2	0.1	800	100	1.81	1.34
FirstFit(HeuristicDegree)	2	0.1	900	100	1.92	1.39
FirstFit(HeuristicDegree)	2	0.1	1000	100	1.87	1.44
FirstFit(HeuristicDegree)	3	0.1	50	100	1.68	0.4
FirstFit(HeuristicDegree)	3	0.1	100	100	2.13	0.57
FirstFit(HeuristicDegree)	3	0.1	150	100	2.49	0.71
FirstFit(HeuristicDegree)	3	0.1	200	100	2.56	0.91
FirstFit(HeuristicDegree)	3	0.1	300	100	3.15	1.25
FirstFit(HeuristicDegree)	3	0.1	400	100	3.63	1.24
FirstFit(HeuristicDegree)	3	0.1	500	100	3.74	1.46
FirstFit(HeuristicDegree)	3	0.1	600	100	3.82	1.64
FirstFit(HeuristicDegree)	3	0.1	700	100	4.42	1.77
FirstFit(HeuristicDegree)	3	0.1	800	100	4.51	1.87
FirstFit(HeuristicDegree)	3	0.1	900	100	4.42	2.12
FirstFit(HeuristicDegree)	3	0.1	1000	100	5.01	1.89
FirstFit(HeuristicDegree)	4	0.1	50	100	1.65	0.19
FirstFit(HeuristicDegree)	4	0.1	100	100	2.11	0.27
FirstFit(HeuristicDegree)	4	0.1	150	100	2.66	0.24
FirstFit(HeuristicDegree)	4	0.1	200	100	2.98	0.38
FirstFit(HeuristicDegree)	4	0.1	300	100	3.68	0.53
FirstFit(HeuristicDegree)	4	0.1	400	100	4.25	0.58
FirstFit(HeuristicDegree)	4	0.1	500	100	4.8	0.65
FirstFit(HeuristicDegree)	4	0.1	600	100	5.09	1.08
FirstFit(HeuristicDegree)	4	0.1	700	100	5.75	0.92
FirstFit(HeuristicDegree)	4	0.1	800	100	6.03	0.91
FirstFit(HeuristicDegree)	4	0.1	900	100	6.37	1.22
FirstFit(HeuristicDegree)	4	0.1	1000	100	6.95	1.05
FirstFit(HeuristicDegree)	2	0.2	50	100	1.07	0.21

Continued on next page

Algorithm	k	p	n	N	$\overline{\rho(Alg)}$	$SD(\rho(Alg))$
FirstFit(HeuristicDegree)	2	0.2	100	100	1.11	0.28
FirstFit(HeuristicDegree)	2	0.2	150	100	1.14	0.37
FirstFit(HeuristicDegree)	2	0.2	200	100	1.12	0.35
FirstFit(HeuristicDegree)	2	0.2	300	100	1.2	0.51
FirstFit(HeuristicDegree)	2	0.2	400	100	1.15	0.36
FirstFit(HeuristicDegree)	2	0.2	500	100	1.2	0.49
FirstFit(HeuristicDegree)	2	0.2	600	100	1.2	0.57
FirstFit(HeuristicDegree)	2	0.2	700	100	1.21	0.46
FirstFit(HeuristicDegree)	2	0.2	800	100	1.21	0.4
FirstFit(HeuristicDegree)	2	0.2	900	100	1.18	0.43
FirstFit(HeuristicDegree)	2	0.2	1000	100	1.36	0.65
FirstFit(HeuristicDegree)	3	0.2	50	100	1.38	0.52
FirstFit(HeuristicDegree)	3	0.2	100	100	1.52	0.65
FirstFit(HeuristicDegree)	3	0.2	150	100	1.67	0.76
FirstFit(HeuristicDegree)	3	0.2	200	100	1.53	0.78
FirstFit(HeuristicDegree)	3	0.2	300	100	1.6	0.8
FirstFit(HeuristicDegree)	3	0.2	400	100	1.73	0.99
FirstFit(HeuristicDegree)	3	0.2	500	100	1.88	1.05
FirstFit(HeuristicDegree)	3	0.2	600	100	1.8	1.06
FirstFit(HeuristicDegree)	3	0.2	700	100	1.7	0.97
FirstFit(HeuristicDegree)	3	0.2	800	100	1.82	1.04
FirstFit(HeuristicDegree)	3	0.2	900	100	1.94	1.34
FirstFit(HeuristicDegree)	3	0.2	1000	100	1.99	1.28
FirstFit(HeuristicDegree)	4	0.2	50	100	1.62	0.4
FirstFit(HeuristicDegree)	4	0.2	100	100	2.01	0.65
FirstFit(HeuristicDegree)	4	0.2	150	100	2.26	0.84
FirstFit(HeuristicDegree)	4	0.2	200	100	2.47	1.06
FirstFit(HeuristicDegree)	4	0.2	300	100	2.73	1.19
FirstFit(HeuristicDegree)	4	0.2	400	100	2.84	1.35
FirstFit(HeuristicDegree)	4	0.2	500	100	3.3	1.68
FirstFit(HeuristicDegree)	4	0.2	600	100	2.89	1.7
FirstFit(HeuristicDegree)	4	0.2	700	100	3.3	1.7
FirstFit(HeuristicDegree)	4	0.2	800	100	3.59	1.9
FirstFit(HeuristicDegree)	4	0.2	900	100	3.56	2.06
FirstFit(HeuristicDegree)	4	0.2	1000	100	3.41	2.03
FirstFit(HeuristicDegree)	2	0.3	50	100	1.03	0.13
FirstFit(HeuristicDegree)	2	0.3	100	100	1.02	0.11
FirstFit(HeuristicDegree)	2	0.3	150	100	1.08	0.2
FirstFit(HeuristicDegree)	2	0.3	200	100	1.04	0.17
FirstFit(HeuristicDegree)	2	0.3	300	100	1.08	0.2
FirstFit(HeuristicDegree)	2	0.3	400	100	1.1	0.3
FirstFit(HeuristicDegree)	2	0.3	500	100	1.05	0.14
FirstFit(HeuristicDegree)	2	0.3	600	100	1.08	0.25
FirstFit(HeuristicDegree)	2	0.3	700	100	1.05	0.17
FirstFit(HeuristicDegree)	2	0.3	800	100	1.12	0.3

Continued on next page

Algorithm	k	p	n	N	$\overline{\rho(Alg)}$	$SD(\rho(Alg))$
FirstFit(HeuristicDegree)	2	0.3	900	100	1.08	0.24
FirstFit(HeuristicDegree)	2	0.3	1000	100	1.05	0.22
FirstFit(HeuristicDegree)	3	0.3	50	100	1.12	0.29
FirstFit(HeuristicDegree)	3	0.3	100	100	1.23	0.32
FirstFit(HeuristicDegree)	3	0.3	150	100	1.24	0.46
FirstFit(HeuristicDegree)	3	0.3	200	100	1.18	0.31
FirstFit(HeuristicDegree)	3	0.3	300	100	1.2	0.34
FirstFit(HeuristicDegree)	3	0.3	400	100	1.21	0.33
FirstFit(HeuristicDegree)	3	0.3	500	100	1.27	0.43
FirstFit(HeuristicDegree)	3	0.3	600	100	1.23	0.42
FirstFit(HeuristicDegree)	3	0.3	700	100	1.19	0.27
FirstFit(HeuristicDegree)	3	0.3	800	100	1.2	0.34
FirstFit(HeuristicDegree)	3	0.3	900	100	1.31	0.45
FirstFit(HeuristicDegree)	3	0.3	1000	100	1.22	0.33
FirstFit(HeuristicDegree)	4	0.3	50	100	1.38	0.38
FirstFit(HeuristicDegree)	4	0.3	100	100	1.38	0.53
FirstFit(HeuristicDegree)	4	0.3	150	100	1.44	0.57
FirstFit(HeuristicDegree)	4	0.3	200	100	1.48	0.6
FirstFit(HeuristicDegree)	4	0.3	300	100	1.55	0.6
FirstFit(HeuristicDegree)	4	0.3	400	100	1.52	0.58
FirstFit(HeuristicDegree)	4	0.3	500	100	1.5	0.64
FirstFit(HeuristicDegree)	4	0.3	600	100	1.49	0.57
FirstFit(HeuristicDegree)	4	0.3	700	100	1.68	0.82
FirstFit(HeuristicDegree)	4	0.3	800	100	1.59	0.61
FirstFit(HeuristicDegree)	4	0.3	900	100	1.55	0.53
FirstFit(HeuristicDegree)	4	0.3	1000	100	1.65	0.7

Table 4: BatchDegree: degree-ordered batches with explicit batch size and density p .

Algorithm	k	p	BatchSize	n	N	$\overline{\rho(Alg)}$	$SD(\rho(Alg))$
FirstFitBatchDegree(size=10)	2	0.05	10.0	50	100	1.94	0.33
FirstFitBatchDegree(size=10)	2	0.05	10.0	100	100	2.47	0.47
FirstFitBatchDegree(size=10)	2	0.05	10.0	150	100	2.83	0.53
FirstFitBatchDegree(size=10)	2	0.05	10.0	200	100	3.25	0.61
FirstFitBatchDegree(size=10)	2	0.05	10.0	300	100	3.6	0.8
FirstFitBatchDegree(size=10)	2	0.05	10.0	400	100	4.13	0.89
FirstFitBatchDegree(size=10)	2	0.05	10.0	500	100	4.4	0.87
FirstFitBatchDegree(size=10)	2	0.05	10.0	600	100	4.54	0.97
FirstFitBatchDegree(size=10)	2	0.05	10.0	700	100	4.8	1.14
FirstFitBatchDegree(size=10)	2	0.05	10.0	800	100	5.1	1.08
FirstFitBatchDegree(size=10)	2	0.05	10.0	900	100	4.95	1.21
FirstFitBatchDegree(size=10)	2	0.05	10.0	1000	100	5.08	1.32
FirstFitBatchDegree(size=10)	3	0.05	10.0	50	100	1.82	0.18

Continued on next page

Algorithm	<i>k</i>	<i>p</i>	BatchSize	<i>n</i>	<i>N</i>	$\overline{\rho(Alg)}$	$SD(\rho(Alg))$
FirstFitBatchDegree(size=10)	3	0.05	10.0	100	100	2.28	0.17
FirstFitBatchDegree(size=10)	3	0.05	10.0	150	100	2.74	0.2
FirstFitBatchDegree(size=10)	3	0.05	10.0	200	100	3.1	0.2
FirstFitBatchDegree(size=10)	3	0.05	10.0	300	100	3.76	0.25
FirstFitBatchDegree(size=10)	3	0.05	10.0	400	100	4.38	0.25
FirstFitBatchDegree(size=10)	3	0.05	10.0	500	100	4.84	0.3
FirstFitBatchDegree(size=10)	3	0.05	10.0	600	100	5.33	0.3
FirstFitBatchDegree(size=10)	3	0.05	10.0	700	100	5.76	0.4
FirstFitBatchDegree(size=10)	3	0.05	10.0	800	100	6.27	0.36
FirstFitBatchDegree(size=10)	3	0.05	10.0	900	100	6.57	0.52
FirstFitBatchDegree(size=10)	3	0.05	10.0	1000	100	6.98	0.56
FirstFitBatchDegree(size=10)	4	0.05	10.0	50	100	1.61	0.15
FirstFitBatchDegree(size=10)	4	0.05	10.0	100	100	2.01	0.12
FirstFitBatchDegree(size=10)	4	0.05	10.0	150	100	2.35	0.14
FirstFitBatchDegree(size=10)	4	0.05	10.0	200	100	2.68	0.13
FirstFitBatchDegree(size=10)	4	0.05	10.0	300	100	3.23	0.14
FirstFitBatchDegree(size=10)	4	0.05	10.0	400	100	3.76	0.15
FirstFitBatchDegree(size=10)	4	0.05	10.0	500	100	4.23	0.17
FirstFitBatchDegree(size=10)	4	0.05	10.0	600	100	4.69	0.17
FirstFitBatchDegree(size=10)	4	0.05	10.0	700	100	5.16	0.14
FirstFitBatchDegree(size=10)	4	0.05	10.0	800	100	5.54	0.22
FirstFitBatchDegree(size=10)	4	0.05	10.0	900	100	6	0.17
FirstFitBatchDegree(size=10)	4	0.05	10.0	1000	100	6.37	0.18
FirstFitBatchDegree(size=50)	2	0.05	50.0	50	100	1.37	0.42
FirstFitBatchDegree(size=50)	2	0.05	50.0	100	100	2.14	0.58
FirstFitBatchDegree(size=50)	2	0.05	50.0	150	100	2.52	0.79
FirstFitBatchDegree(size=50)	2	0.05	50.0	200	100	2.99	0.78
FirstFitBatchDegree(size=50)	2	0.05	50.0	300	100	3.31	0.9
FirstFitBatchDegree(size=50)	2	0.05	50.0	400	100	3.78	0.99
FirstFitBatchDegree(size=50)	2	0.05	50.0	500	100	3.95	1.35
FirstFitBatchDegree(size=50)	2	0.05	50.0	600	100	4.19	1.23
FirstFitBatchDegree(size=50)	2	0.05	50.0	700	100	4.3	1.49
FirstFitBatchDegree(size=50)	2	0.05	50.0	800	100	4.67	1.39
FirstFitBatchDegree(size=50)	2	0.05	50.0	900	100	4.89	1.38
FirstFitBatchDegree(size=50)	2	0.05	50.0	1000	100	4.7	1.35
FirstFitBatchDegree(size=50)	3	0.05	50.0	50	100	1.59	0.19
FirstFitBatchDegree(size=50)	3	0.05	50.0	100	100	2.16	0.22
FirstFitBatchDegree(size=50)	3	0.05	50.0	150	100	2.59	0.23
FirstFitBatchDegree(size=50)	3	0.05	50.0	200	100	3.03	0.25
FirstFitBatchDegree(size=50)	3	0.05	50.0	300	100	3.71	0.26
FirstFitBatchDegree(size=50)	3	0.05	50.0	400	100	4.28	0.28
FirstFitBatchDegree(size=50)	3	0.05	50.0	500	100	4.79	0.3
FirstFitBatchDegree(size=50)	3	0.05	50.0	600	100	5.21	0.4
FirstFitBatchDegree(size=50)	3	0.05	50.0	700	100	5.76	0.35
FirstFitBatchDegree(size=50)	3	0.05	50.0	800	100	6.11	0.5

Continued on next page

Algorithm	k	p	BatchSize	n	N	$\overline{\rho(Alg)}$	$SD(\rho(Alg))$
FirstFitBatchDegree(size=50)	3	0.05	50.0	900	100	6.46	0.45
FirstFitBatchDegree(size=50)	3	0.05	50.0	1000	100	6.98	0.53
FirstFitBatchDegree(size=50)	4	0.05	50.0	50	100	1.46	0.12
FirstFitBatchDegree(size=50)	4	0.05	50.0	100	100	1.89	0.14
FirstFitBatchDegree(size=50)	4	0.05	50.0	150	100	2.29	0.14
FirstFitBatchDegree(size=50)	4	0.05	50.0	200	100	2.6	0.14
FirstFitBatchDegree(size=50)	4	0.05	50.0	300	100	3.2	0.14
FirstFitBatchDegree(size=50)	4	0.05	50.0	400	100	3.72	0.16
FirstFitBatchDegree(size=50)	4	0.05	50.0	500	100	4.18	0.17
FirstFitBatchDegree(size=50)	4	0.05	50.0	600	100	4.66	0.15
FirstFitBatchDegree(size=50)	4	0.05	50.0	700	100	5.11	0.16
FirstFitBatchDegree(size=50)	4	0.05	50.0	800	100	5.51	0.18
FirstFitBatchDegree(size=50)	4	0.05	50.0	900	100	5.91	0.19
FirstFitBatchDegree(size=50)	4	0.05	50.0	1000	100	6.31	0.19
FirstFitBatchDegree(size=10)	2	0.1	10.0	50	100	1.93	0.55
FirstFitBatchDegree(size=10)	2	0.1	10.0	100	100	2.5	0.74
FirstFitBatchDegree(size=10)	2	0.1	10.0	150	100	2.63	0.99
FirstFitBatchDegree(size=10)	2	0.1	10.0	200	100	2.87	1.04
FirstFitBatchDegree(size=10)	2	0.1	10.0	300	100	3.06	1.22
FirstFitBatchDegree(size=10)	2	0.1	10.0	400	100	3.3	1.41
FirstFitBatchDegree(size=10)	2	0.1	10.0	500	100	3.49	1.61
FirstFitBatchDegree(size=10)	2	0.1	10.0	600	100	3.08	1.43
FirstFitBatchDegree(size=10)	2	0.1	10.0	700	100	3.35	1.7
FirstFitBatchDegree(size=10)	2	0.1	10.0	800	100	3.87	1.71
FirstFitBatchDegree(size=10)	2	0.1	10.0	900	100	3.88	1.93
FirstFitBatchDegree(size=10)	2	0.1	10.0	1000	100	3.6	1.82
FirstFitBatchDegree(size=10)	3	0.1	10.0	50	100	2.09	0.24
FirstFitBatchDegree(size=10)	3	0.1	10.0	100	100	2.75	0.32
FirstFitBatchDegree(size=10)	3	0.1	10.0	150	100	3.27	0.47
FirstFitBatchDegree(size=10)	3	0.1	10.0	200	100	3.78	0.52
FirstFitBatchDegree(size=10)	3	0.1	10.0	300	100	4.34	0.72
FirstFitBatchDegree(size=10)	3	0.1	10.0	400	100	5	0.77
FirstFitBatchDegree(size=10)	3	0.1	10.0	500	100	5.56	1.06
FirstFitBatchDegree(size=10)	3	0.1	10.0	600	100	6.07	1.01
FirstFitBatchDegree(size=10)	3	0.1	10.0	700	100	6.46	1.13
FirstFitBatchDegree(size=10)	3	0.1	10.0	800	100	6.57	1.35
FirstFitBatchDegree(size=10)	3	0.1	10.0	900	100	7.12	1.41
FirstFitBatchDegree(size=10)	3	0.1	10.0	1000	100	7.32	1.5
FirstFitBatchDegree(size=10)	4	0.1	10.0	50	100	1.84	0.16
FirstFitBatchDegree(size=10)	4	0.1	10.0	100	100	2.47	0.19
FirstFitBatchDegree(size=10)	4	0.1	10.0	150	100	3.01	0.2
FirstFitBatchDegree(size=10)	4	0.1	10.0	200	100	3.51	0.22
FirstFitBatchDegree(size=10)	4	0.1	10.0	300	100	4.37	0.28
FirstFitBatchDegree(size=10)	4	0.1	10.0	400	100	5.16	0.3
FirstFitBatchDegree(size=10)	4	0.1	10.0	500	100	5.75	0.43

Continued on next page

Algorithm	<i>k</i>	<i>p</i>	BatchSize	<i>n</i>	<i>N</i>	$\overline{\rho(Alg)}$	$SD(\rho(Alg))$
FirstFitBatchDegree(size=10)	4	0.1	10.0	600	100	6.34	0.49
FirstFitBatchDegree(size=10)	4	0.1	10.0	700	100	6.84	0.54
FirstFitBatchDegree(size=10)	4	0.1	10.0	800	100	7.45	0.78
FirstFitBatchDegree(size=10)	4	0.1	10.0	900	100	8.01	0.7
FirstFitBatchDegree(size=10)	4	0.1	10.0	1000	100	8.48	0.91
FirstFitBatchDegree(size=50)	2	0.1	50.0	50	100	1.35	0.51
FirstFitBatchDegree(size=50)	2	0.1	50.0	100	100	1.79	0.85
FirstFitBatchDegree(size=50)	2	0.1	50.0	150	100	1.96	0.95
FirstFitBatchDegree(size=50)	2	0.1	50.0	200	100	2.06	1.12
FirstFitBatchDegree(size=50)	2	0.1	50.0	300	100	2.5	1.31
FirstFitBatchDegree(size=50)	2	0.1	50.0	400	100	2.59	1.45
FirstFitBatchDegree(size=50)	2	0.1	50.0	500	100	2.64	1.53
FirstFitBatchDegree(size=50)	2	0.1	50.0	600	100	2.7	1.56
FirstFitBatchDegree(size=50)	2	0.1	50.0	700	100	3.18	1.76
FirstFitBatchDegree(size=50)	2	0.1	50.0	800	100	2.93	1.67
FirstFitBatchDegree(size=50)	2	0.1	50.0	900	100	2.95	1.75
FirstFitBatchDegree(size=50)	2	0.1	50.0	1000	100	3.23	1.93
FirstFitBatchDegree(size=50)	3	0.1	50.0	50	100	1.68	0.4
FirstFitBatchDegree(size=50)	3	0.1	50.0	100	100	2.48	0.47
FirstFitBatchDegree(size=50)	3	0.1	50.0	150	100	3	0.52
FirstFitBatchDegree(size=50)	3	0.1	50.0	200	100	3.53	0.7
FirstFitBatchDegree(size=50)	3	0.1	50.0	300	100	4.42	0.66
FirstFitBatchDegree(size=50)	3	0.1	50.0	400	100	4.78	1.14
FirstFitBatchDegree(size=50)	3	0.1	50.0	500	100	5.33	1.01
FirstFitBatchDegree(size=50)	3	0.1	50.0	600	100	5.59	1.44
FirstFitBatchDegree(size=50)	3	0.1	50.0	700	100	6.04	1.33
FirstFitBatchDegree(size=50)	3	0.1	50.0	800	100	6.49	1.27
FirstFitBatchDegree(size=50)	3	0.1	50.0	900	100	6.69	1.67
FirstFitBatchDegree(size=50)	3	0.1	50.0	1000	100	7.08	1.57
FirstFitBatchDegree(size=50)	4	0.1	50.0	50	100	1.65	0.19
FirstFitBatchDegree(size=50)	4	0.1	50.0	100	100	2.32	0.19
FirstFitBatchDegree(size=50)	4	0.1	50.0	150	100	2.88	0.19
FirstFitBatchDegree(size=50)	4	0.1	50.0	200	100	3.41	0.19
FirstFitBatchDegree(size=50)	4	0.1	50.0	300	100	4.31	0.25
FirstFitBatchDegree(size=50)	4	0.1	50.0	400	100	5.02	0.32
FirstFitBatchDegree(size=50)	4	0.1	50.0	500	100	5.74	0.42
FirstFitBatchDegree(size=50)	4	0.1	50.0	600	100	6.42	0.47
FirstFitBatchDegree(size=50)	4	0.1	50.0	700	100	6.89	0.54
FirstFitBatchDegree(size=50)	4	0.1	50.0	800	100	7.4	0.64
FirstFitBatchDegree(size=50)	4	0.1	50.0	900	100	8.04	0.64
FirstFitBatchDegree(size=50)	4	0.1	50.0	1000	100	8.32	0.84
FirstFitBatchDegree(size=10)	2	0.2	10.0	50	100	1.52	0.58
FirstFitBatchDegree(size=10)	2	0.2	10.0	100	100	1.79	0.85
FirstFitBatchDegree(size=10)	2	0.2	10.0	150	100	1.75	0.9
FirstFitBatchDegree(size=10)	2	0.2	10.0	200	100	1.78	0.75

Continued on next page

Algorithm	k	p	BatchSize	n	N	$\overline{\rho(Alg)}$	$SD(\rho(Alg))$
FirstFitBatchDegree(size=10)	2	0.2	10.0	300	100	1.79	0.87
FirstFitBatchDegree(size=10)	2	0.2	10.0	400	100	1.82	0.86
FirstFitBatchDegree(size=10)	2	0.2	10.0	500	100	1.63	0.76
FirstFitBatchDegree(size=10)	2	0.2	10.0	600	100	1.77	0.9
FirstFitBatchDegree(size=10)	2	0.2	10.0	700	100	2	1.08
FirstFitBatchDegree(size=10)	2	0.2	10.0	800	100	1.89	1.03
FirstFitBatchDegree(size=10)	2	0.2	10.0	900	100	1.7	0.8
FirstFitBatchDegree(size=10)	2	0.2	10.0	1000	100	1.93	1.25
FirstFitBatchDegree(size=10)	3	0.2	10.0	50	100	2.07	0.58
FirstFitBatchDegree(size=10)	3	0.2	10.0	100	100	2.59	0.83
FirstFitBatchDegree(size=10)	3	0.2	10.0	150	100	2.88	0.94
FirstFitBatchDegree(size=10)	3	0.2	10.0	200	100	3.14	1.26
FirstFitBatchDegree(size=10)	3	0.2	10.0	300	100	3.24	1.46
FirstFitBatchDegree(size=10)	3	0.2	10.0	400	100	3.43	1.56
FirstFitBatchDegree(size=10)	3	0.2	10.0	500	100	3.74	1.71
FirstFitBatchDegree(size=10)	3	0.2	10.0	600	100	3.83	1.92
FirstFitBatchDegree(size=10)	3	0.2	10.0	700	100	3.4	1.68
FirstFitBatchDegree(size=10)	3	0.2	10.0	800	100	3.62	1.77
FirstFitBatchDegree(size=10)	3	0.2	10.0	900	100	3.82	1.92
FirstFitBatchDegree(size=10)	3	0.2	10.0	1000	100	3.98	2.25
FirstFitBatchDegree(size=10)	4	0.2	10.0	50	100	2.07	0.34
FirstFitBatchDegree(size=10)	4	0.2	10.0	100	100	2.74	0.49
FirstFitBatchDegree(size=10)	4	0.2	10.0	150	100	3.42	0.65
FirstFitBatchDegree(size=10)	4	0.2	10.0	200	100	3.72	0.86
FirstFitBatchDegree(size=10)	4	0.2	10.0	300	100	4.64	1.04
FirstFitBatchDegree(size=10)	4	0.2	10.0	400	100	4.68	1.51
FirstFitBatchDegree(size=10)	4	0.2	10.0	500	100	5.11	1.64
FirstFitBatchDegree(size=10)	4	0.2	10.0	600	100	5.25	1.84
FirstFitBatchDegree(size=10)	4	0.2	10.0	700	100	5.52	1.84
FirstFitBatchDegree(size=10)	4	0.2	10.0	800	100	6.32	2.05
FirstFitBatchDegree(size=10)	4	0.2	10.0	900	100	6.45	2.03
FirstFitBatchDegree(size=10)	4	0.2	10.0	1000	100	5.81	2.23
FirstFitBatchDegree(size=50)	2	0.2	50.0	50	100	1.07	0.21
FirstFitBatchDegree(size=50)	2	0.2	50.0	100	100	1.17	0.39
FirstFitBatchDegree(size=50)	2	0.2	50.0	150	100	1.34	0.69
FirstFitBatchDegree(size=50)	2	0.2	50.0	200	100	1.3	0.62
FirstFitBatchDegree(size=50)	2	0.2	50.0	300	100	1.45	0.68
FirstFitBatchDegree(size=50)	2	0.2	50.0	400	100	1.44	0.85
FirstFitBatchDegree(size=50)	2	0.2	50.0	500	100	1.6	0.94
FirstFitBatchDegree(size=50)	2	0.2	50.0	600	100	1.4	0.69
FirstFitBatchDegree(size=50)	2	0.2	50.0	700	100	1.45	0.73
FirstFitBatchDegree(size=50)	2	0.2	50.0	800	100	1.39	0.75
FirstFitBatchDegree(size=50)	2	0.2	50.0	900	100	1.65	0.98
FirstFitBatchDegree(size=50)	2	0.2	50.0	1000	100	1.48	0.74
FirstFitBatchDegree(size=50)	3	0.2	50.0	50	100	1.38	0.52

Continued on next page

Algorithm	<i>k</i>	<i>p</i>	BatchSize	<i>n</i>	<i>N</i>	$\overline{\rho(Alg)}$	$SD(\rho(Alg))$
FirstFitBatchDegree(size=50)	3	0.2	50.0	100	100	1.81	0.8
FirstFitBatchDegree(size=50)	3	0.2	50.0	150	100	2.04	0.95
FirstFitBatchDegree(size=50)	3	0.2	50.0	200	100	2.31	1.17
FirstFitBatchDegree(size=50)	3	0.2	50.0	300	100	2.61	1.57
FirstFitBatchDegree(size=50)	3	0.2	50.0	400	100	2.59	1.41
FirstFitBatchDegree(size=50)	3	0.2	50.0	500	100	2.91	1.76
FirstFitBatchDegree(size=50)	3	0.2	50.0	600	100	3.1	1.7
FirstFitBatchDegree(size=50)	3	0.2	50.0	700	100	2.8	1.67
FirstFitBatchDegree(size=50)	3	0.2	50.0	800	100	3.33	2.12
FirstFitBatchDegree(size=50)	3	0.2	50.0	900	100	3.27	1.87
FirstFitBatchDegree(size=50)	3	0.2	50.0	1000	100	2.94	1.79
FirstFitBatchDegree(size=50)	4	0.2	50.0	50	100	1.62	0.4
FirstFitBatchDegree(size=50)	4	0.2	50.0	100	100	2.49	0.57
FirstFitBatchDegree(size=50)	4	0.2	50.0	150	100	3.15	0.8
FirstFitBatchDegree(size=50)	4	0.2	50.0	200	100	3.32	1.13
FirstFitBatchDegree(size=50)	4	0.2	50.0	300	100	4.06	1.16
FirstFitBatchDegree(size=50)	4	0.2	50.0	400	100	4.32	1.54
FirstFitBatchDegree(size=50)	4	0.2	50.0	500	100	4.95	1.93
FirstFitBatchDegree(size=50)	4	0.2	50.0	600	100	5.34	2.01
FirstFitBatchDegree(size=50)	4	0.2	50.0	700	100	5.32	2.01
FirstFitBatchDegree(size=50)	4	0.2	50.0	800	100	5.65	2.33
FirstFitBatchDegree(size=50)	4	0.2	50.0	900	100	5.9	2.33
FirstFitBatchDegree(size=50)	4	0.2	50.0	1000	100	6.41	2.35
FirstFitBatchDegree(size=10)	2	0.3	10.0	50	100	1.31	0.45
FirstFitBatchDegree(size=10)	2	0.3	10.0	100	100	1.26	0.43
FirstFitBatchDegree(size=10)	2	0.3	10.0	150	100	1.39	0.61
FirstFitBatchDegree(size=10)	2	0.3	10.0	200	100	1.41	0.69
FirstFitBatchDegree(size=10)	2	0.3	10.0	300	100	1.35	0.52
FirstFitBatchDegree(size=10)	2	0.3	10.0	400	100	1.2	0.34
FirstFitBatchDegree(size=10)	2	0.3	10.0	500	100	1.34	0.59
FirstFitBatchDegree(size=10)	2	0.3	10.0	600	100	1.33	0.54
FirstFitBatchDegree(size=10)	2	0.3	10.0	700	100	1.3	0.48
FirstFitBatchDegree(size=10)	2	0.3	10.0	800	100	1.3	0.52
FirstFitBatchDegree(size=10)	2	0.3	10.0	900	100	1.4	0.63
FirstFitBatchDegree(size=10)	2	0.3	10.0	1000	100	1.41	0.53
FirstFitBatchDegree(size=10)	3	0.3	10.0	50	100	1.63	0.56
FirstFitBatchDegree(size=10)	3	0.3	10.0	100	100	1.84	0.75
FirstFitBatchDegree(size=10)	3	0.3	10.0	150	100	1.99	0.94
FirstFitBatchDegree(size=10)	3	0.3	10.0	200	100	2.02	0.99
FirstFitBatchDegree(size=10)	3	0.3	10.0	300	100	1.9	0.76
FirstFitBatchDegree(size=10)	3	0.3	10.0	400	100	2.02	0.98
FirstFitBatchDegree(size=10)	3	0.3	10.0	500	100	2.03	0.89
FirstFitBatchDegree(size=10)	3	0.3	10.0	600	100	1.94	0.88
FirstFitBatchDegree(size=10)	3	0.3	10.0	700	100	2.09	1.01
FirstFitBatchDegree(size=10)	3	0.3	10.0	800	100	1.9	0.93

Continued on next page

Algorithm	<i>k</i>	<i>p</i>	BatchSize	<i>n</i>	<i>N</i>	$\overline{\rho(Alg)}$	$SD(\rho(Alg))$
FirstFitBatchDegree(size=10)	3	0.3	10.0	900	100	2.17	1.16
FirstFitBatchDegree(size=10)	3	0.3	10.0	1000	100	2.1	0.94
FirstFitBatchDegree(size=10)	4	0.3	10.0	50	100	1.85	0.49
FirstFitBatchDegree(size=10)	4	0.3	10.0	100	100	2.34	0.74
FirstFitBatchDegree(size=10)	4	0.3	10.0	150	100	2.55	0.97
FirstFitBatchDegree(size=10)	4	0.3	10.0	200	100	2.59	1.14
FirstFitBatchDegree(size=10)	4	0.3	10.0	300	100	2.79	1.1
FirstFitBatchDegree(size=10)	4	0.3	10.0	400	100	2.65	1.06
FirstFitBatchDegree(size=10)	4	0.3	10.0	500	100	2.93	1.33
FirstFitBatchDegree(size=10)	4	0.3	10.0	600	100	2.86	1.23
FirstFitBatchDegree(size=10)	4	0.3	10.0	700	100	2.92	1.38
FirstFitBatchDegree(size=10)	4	0.3	10.0	800	100	3.09	1.53
FirstFitBatchDegree(size=10)	4	0.3	10.0	900	100	3.03	1.52
FirstFitBatchDegree(size=10)	4	0.3	10.0	1000	100	2.98	1.34
FirstFitBatchDegree(size=50)	2	0.3	50.0	50	100	1.03	0.13
FirstFitBatchDegree(size=50)	2	0.3	50.0	100	100	1.04	0.14
FirstFitBatchDegree(size=50)	2	0.3	50.0	150	100	1.1	0.39
FirstFitBatchDegree(size=50)	2	0.3	50.0	200	100	1.13	0.31
FirstFitBatchDegree(size=50)	2	0.3	50.0	300	100	1.15	0.36
FirstFitBatchDegree(size=50)	2	0.3	50.0	400	100	1.14	0.42
FirstFitBatchDegree(size=50)	2	0.3	50.0	500	100	1.17	0.45
FirstFitBatchDegree(size=50)	2	0.3	50.0	600	100	1.17	0.39
FirstFitBatchDegree(size=50)	2	0.3	50.0	700	100	1.1	0.25
FirstFitBatchDegree(size=50)	2	0.3	50.0	800	100	1.14	0.33
FirstFitBatchDegree(size=50)	2	0.3	50.0	900	100	1.16	0.34
FirstFitBatchDegree(size=50)	2	0.3	50.0	1000	100	1.16	0.31
FirstFitBatchDegree(size=50)	3	0.3	50.0	50	100	1.12	0.29
FirstFitBatchDegree(size=50)	3	0.3	50.0	100	100	1.36	0.46
FirstFitBatchDegree(size=50)	3	0.3	50.0	150	100	1.43	0.61
FirstFitBatchDegree(size=50)	3	0.3	50.0	200	100	1.31	0.44
FirstFitBatchDegree(size=50)	3	0.3	50.0	300	100	1.46	0.68
FirstFitBatchDegree(size=50)	3	0.3	50.0	400	100	1.56	0.78
FirstFitBatchDegree(size=50)	3	0.3	50.0	500	100	1.63	0.8
FirstFitBatchDegree(size=50)	3	0.3	50.0	600	100	1.61	0.79
FirstFitBatchDegree(size=50)	3	0.3	50.0	700	100	1.64	0.93
FirstFitBatchDegree(size=50)	3	0.3	50.0	800	100	1.51	0.63
FirstFitBatchDegree(size=50)	3	0.3	50.0	900	100	1.61	0.67
FirstFitBatchDegree(size=50)	3	0.3	50.0	1000	100	1.58	0.81
FirstFitBatchDegree(size=50)	4	0.3	50.0	50	100	1.38	0.38
FirstFitBatchDegree(size=50)	4	0.3	50.0	100	100	1.66	0.69
FirstFitBatchDegree(size=50)	4	0.3	50.0	150	100	1.95	0.8
FirstFitBatchDegree(size=50)	4	0.3	50.0	200	100	2.16	1.02
FirstFitBatchDegree(size=50)	4	0.3	50.0	300	100	2.21	1.11
FirstFitBatchDegree(size=50)	4	0.3	50.0	400	100	2.17	1.09
FirstFitBatchDegree(size=50)	4	0.3	50.0	500	100	2.33	1.2

Continued on next page

Algorithm	k	p	BatchSize	n	N	$\overline{\rho(Alg)}$	$SD(\rho(Alg))$
FirstFitBatchDegree(size=50)	4	0.3	50.0	600	100	2.61	1.36
FirstFitBatchDegree(size=50)	4	0.3	50.0	700	100	2.2	1.18
FirstFitBatchDegree(size=50)	4	0.3	50.0	800	100	2.09	1.07
FirstFitBatchDegree(size=50)	4	0.3	50.0	900	100	2.34	1.18
FirstFitBatchDegree(size=50)	4	0.3	50.0	1000	100	2.71	1.53

6 Analysis

We now analyse outcomes for the two simulation tracks. Each subsection first presents raw behavior (grouped figures) followed by model-fitting interpretation. Competitive ratio $\rho = \text{used colors}/k$ closer to 1 indicates tighter adherence to the target palette.

6.1 Simulation I (FirstFit and CBIP)

Grouped FirstFit performance A composite grid (Figure 1) summarises $k \in \{2, 3, 4\}$ across densities $p \in \{0.05, 0.1, 0.2, 0.3\}$.

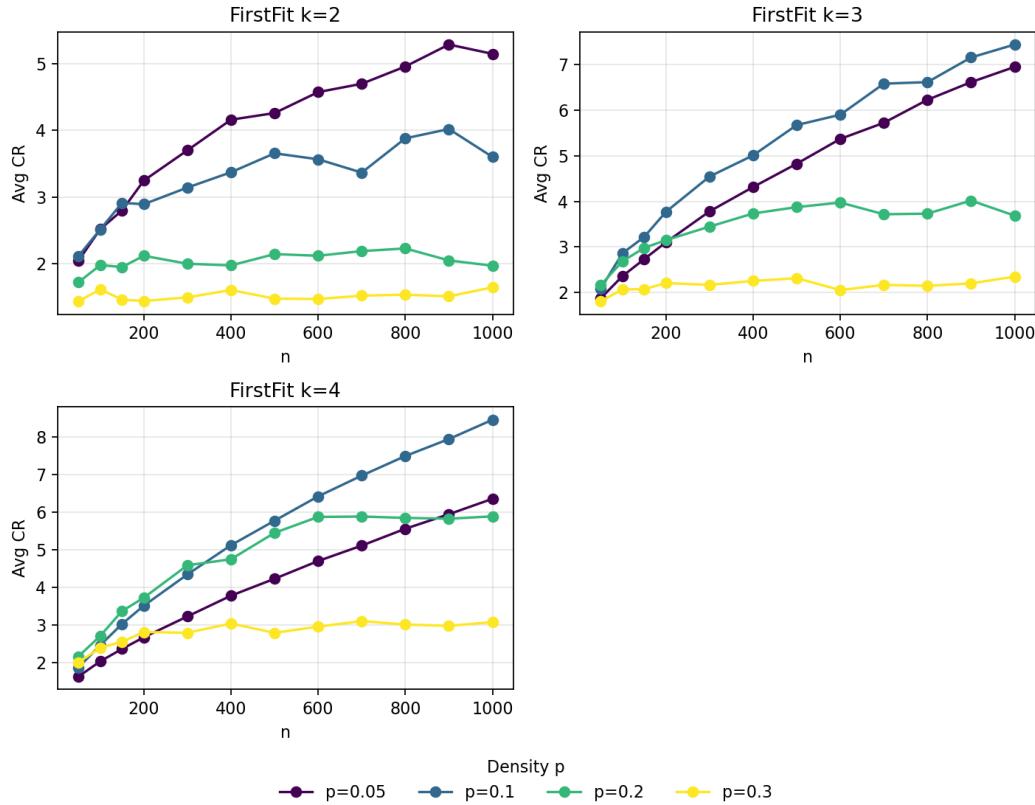


Figure 1: FirstFit competitive ratio vs n for $k = 2, 3, 4$ across densities p .

FirstFit grid: In this dataset, several curves show that higher density p corresponds to *lower* competitive ratios over large ranges of n , and lines can cross. This non-monotonic effect suggests

that denser instances sometimes create more regular neighbourhoods where FirstFit reuses colours efficiently, whereas sparser instances trigger extra colours earlier due to irregular conflicts. The $k = 2$ panels stay near $\rho \approx 1$ (consistent with near-bipartite behaviour). For $k = 3$ and $k = 4$, variability increases and the ordering by p is not stable across all n ; both increasing and decreasing segments with p appear. Overall, the impact of p on FirstFit is **non-monotonic** in our results, with several cases where larger p yields lower ρ .

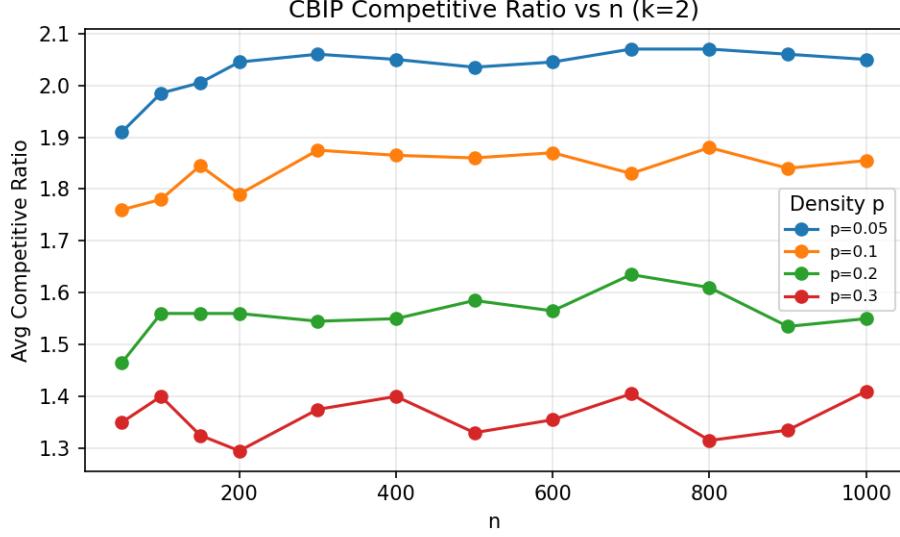


Figure 2: CBIP competitive ratio vs n for $k = 2$ across densities p (nearly flat).

CBIP ($k = 2$): CBIP traces almost constant $\rho \approx 1$ across n and all p tested. Its BFS bipartition guarantees minimal colors required for bipartite substructure revealed so far. Variance is negligible compared to FirstFit.

Why is k restricted to 2 in CBIP?

Bipartition offers a linear-time certificate and immediate 2-coloring. Extending CBIP beyond bipartite graphs would require online k -coloring (NP-complete for $k \geq 3$), removing tractable structural guarantees and making per-arrival component re-coloring infeasible at our scales. Hence we restrict CBIP reporting to $k = 2$.

Model fitting To characterise growth, we fitted $\log n$, \sqrt{n} , and quadratic models to mean ρ vs n per (k, p) . For FirstFit, $\log n$ often suffices at low p (sublinear growth of conflicts), while higher p shifts preference toward \sqrt{n} or quadratic where acceleration of conflicts appears. CBIP exhibits near-constant behavior, with $a + b \log n$ degenerating to an almost flat line and lowest RMSE.

Best-fit summaries

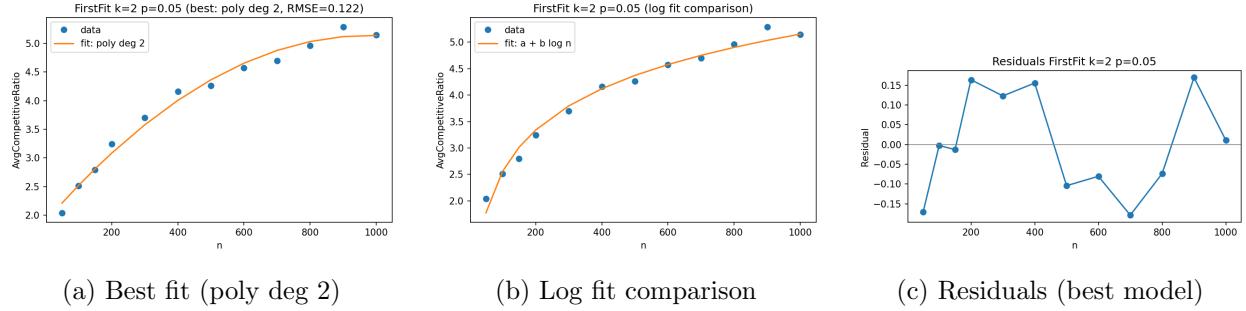


Figure 3: FirstFit $k = 2, p = 0.05$: mild sublinear growth. Quadratic (degree-2) model selected (lowest RMSE); $\log(n)$ fit shows similar gentle curvature with slightly higher error.

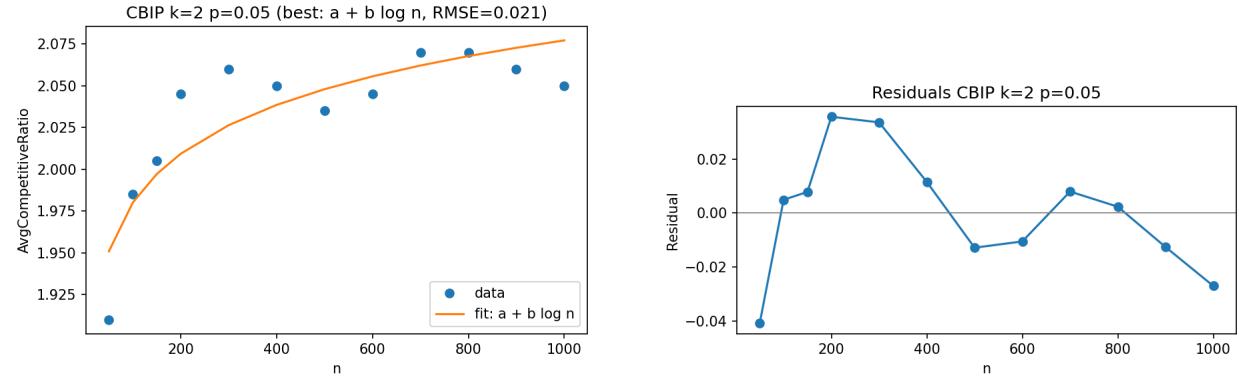


Figure 4: CBIP $k = 2, p = 0.05$: near-flat residuals confirm structural optimality.

6.2 Simulation II (HeuristicDegree and BatchDegree)

Grouped comparison figures by density For each k , we present a 2×2 grid of $p \in \{0.05, 0.1, 0.2, 0.3\}$; each subplot compares FirstFit, HeuristicDegree, and BatchDegree variants over n .

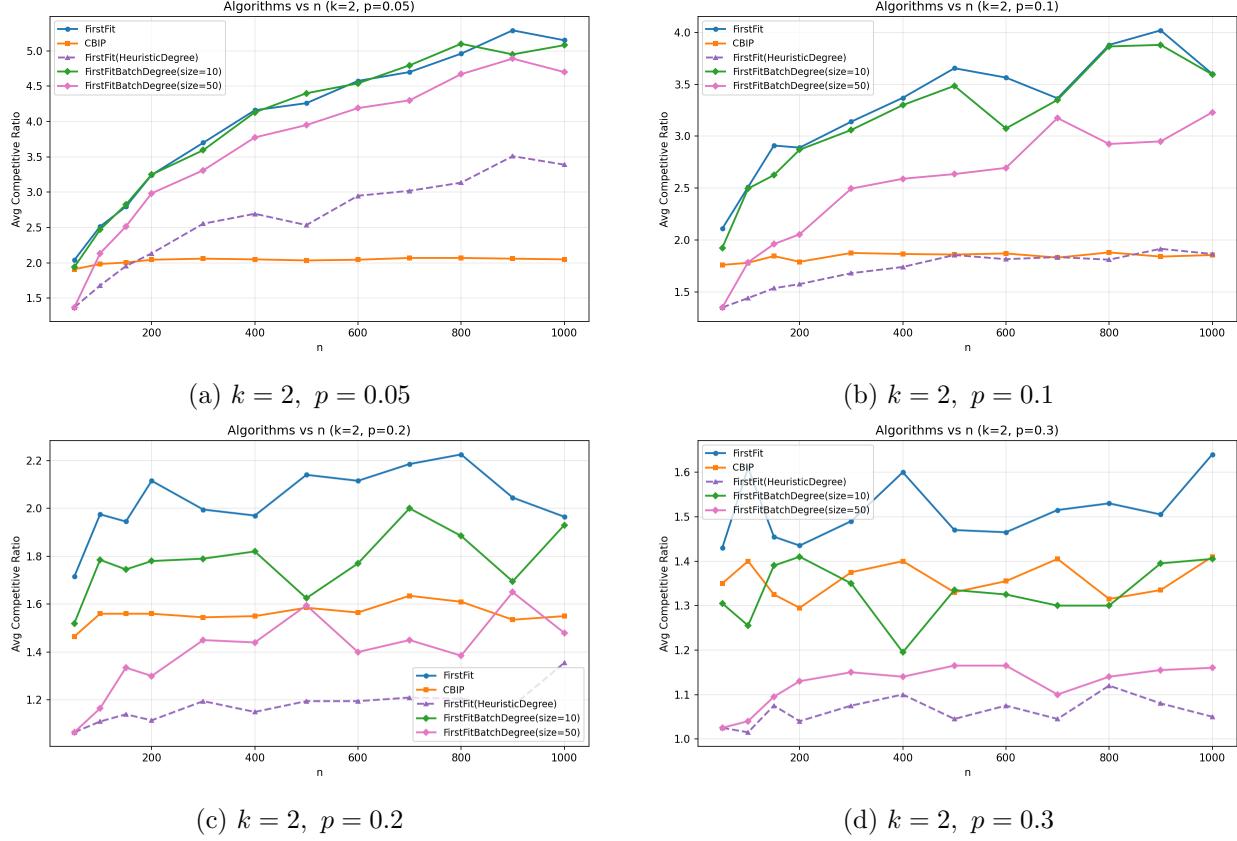


Figure 5: Simulation II grouped comparison for $k = 2$: FirstFit vs HeuristicDegree vs BatchDegree across densities.

Analysis ($k = 2$): HeuristicDegree reduces ρ relative to FirstFit across all densities and remains the strongest variant. **However, BatchDegree still improves upon raw FirstFit at higher densities when the batch size is larger.** For example, at $p = 0.2$ and $n = 1000$ FirstFit attains $\rho \approx 1.97$ while BatchDegree($t = 50$) lowers this to ≈ 1.48 (25% reduction); at $p = 0.3$ and $n = 1000$ FirstFit is ≈ 1.64 versus ≈ 1.16 for BatchDegree($t = 50$) (29% reduction). Smaller batches ($t = 10$) yield weaker or inconsistent gains. Thus the ordering hierarchy is HeuristicDegree < BatchDegree($t = 50$) < FirstFit (in ρ) for dense settings, while BatchDegree may revert closer to FirstFit at lower p .

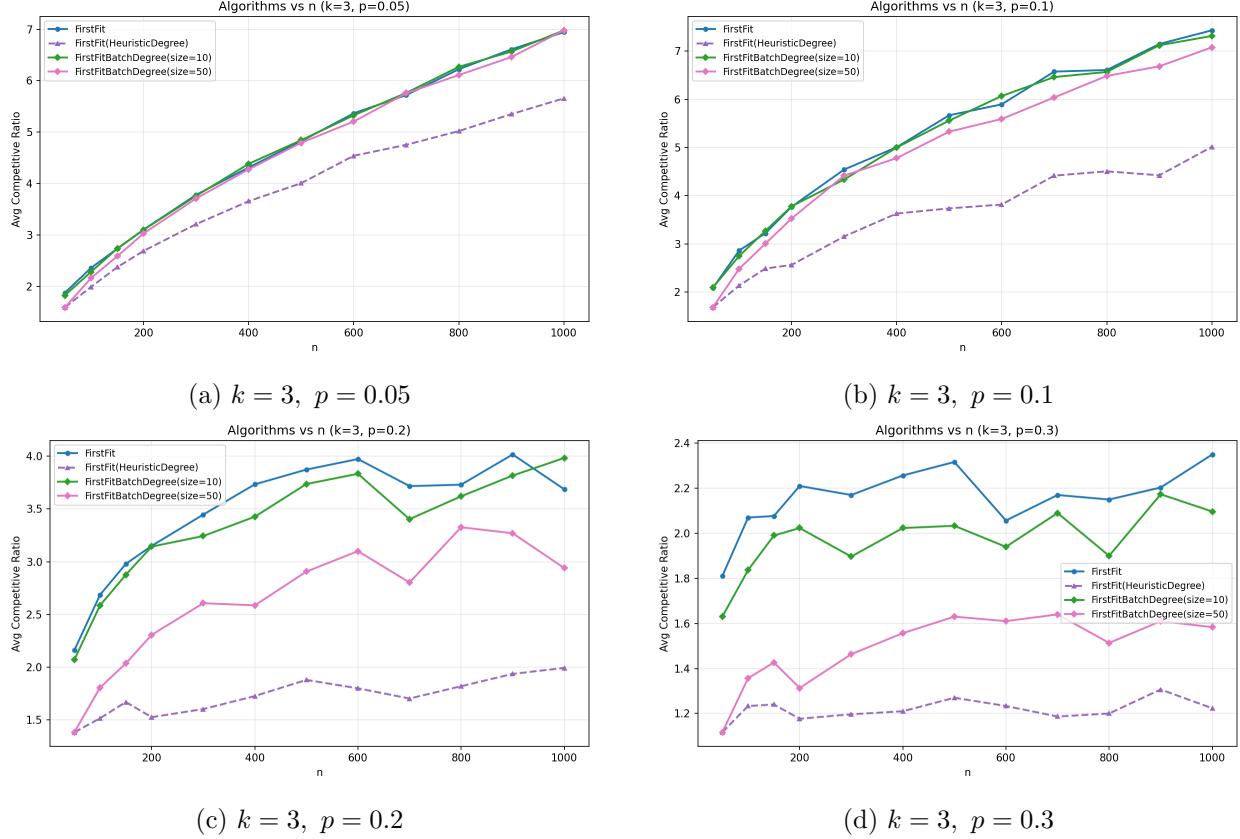


Figure 6: Simulation II grouped comparison for $k = 3$: FirstFit vs HeuristicDegree vs BatchDegree across densities.

Analysis ($k = 3$): HeuristicDegree remains the strongest performer, but larger batches again provide **meaningful improvement over FirstFit in dense regimes**. At $p = 0.2$, $n = 1000$ FirstFit reaches $\rho \approx 3.69$ while BatchDegree($t = 50$) reduces this to ≈ 2.94 (20% reduction); at $p = 0.3$, $n = 1000$ FirstFit is ≈ 2.35 versus ≈ 1.58 for BatchDegree($t = 50$) (33% reduction). BatchDegree($t = 10$) offers smaller or unstable gains. Ordering relationship for dense cases: HeuristicDegree < BatchDegree($t = 50$) < FirstFit. For sparser graphs the BatchDegree curves can drift upward toward FirstFit, reflecting diminished benefit when local batches see fewer high-degree conflicts to exploit.

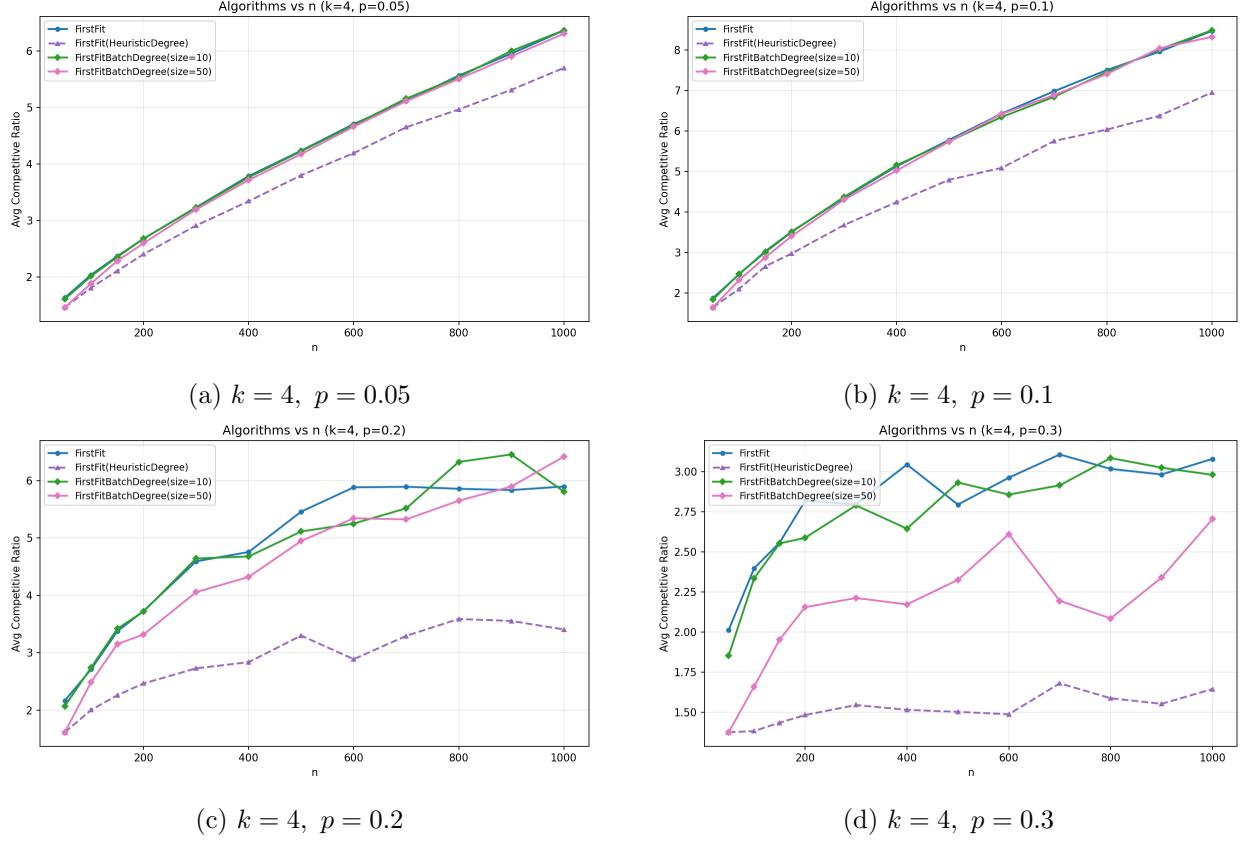


Figure 7: Simulation II grouped comparison for $k = 4$: FirstFit vs HeuristicDegree vs BatchDegree across densities.

Analysis ($k = 4$): HeuristicDegree again delivers the lowest ratios. BatchDegree($t = 50$) shows **modest** improvement over FirstFit at the highest density ($p = 0.3$, $n = 1000$: ρ drops from ≈ 3.08 to ≈ 2.71 , 12% reduction) while at $p = 0.2$ improvements do not materialise (BatchDegree slightly worse than FirstFit for large n). This indicates diminishing returns of limited-window reordering as k grows: global sorting preserves advantages, whereas intra-batch ordering may not sufficiently coordinate color reuse across partitions.

7 Conclusion

This project implemented and rigorously evaluated online graph coloring baselines (FirstFit, CBIP) and degree-based ordering heuristics (HeuristicDegree, BatchDegree) over reproducible, cached datasets spanning n , k , and p . We standardised algorithm descriptions, ensured correctness via unit tests, and produced a coherent analysis pipeline (figures, fits, and summaries).

Key findings:

- **FirstFit behavior:** Competitive ratio trends with density p are non-monotonic; higher p can yield lower ratios via more regular neighbourhoods enabling color reuse. Growth with n is generally mild to sublinear.
- **CBIP ($k=2$):** Near-optimal ($\rho \approx 1$) and highly stable across n and p , confirming bipartite structural leverage.

- **Ordering pays off (nuanced):** HeuristicDegree is consistently best. BatchDegree with larger batches ($t = 50$) still achieves substantial reductions vs raw FirstFit in dense settings (e.g., $\approx 25\%$ for $k = 2, p = 0.2$; $\approx 33\%$ for $k = 3, p = 0.3$; $\approx 12\%$ for $k = 4, p = 0.3$), though it does not surpass the global ordering and may revert toward FirstFit in sparser cases.
- **Model fits:** Simple forms (log, sqrt, quadratic) capture mild growth; quadratic often attains the lowest error while log remains a close, interpretable alternative.

Overall, our efforts produced a clean, reproducible framework and empirical evidence that strategic ordering, particularly in batched form, meaningfully improves online coloring quality, while CBIP remains the gold standard for bipartite inputs.

8 Team Work Distribution

List tasks per member (generator, algorithms, experiments, analysis, writing, QA).

References

- [1] Antonios Antoniadis, Hajo Broersma, and Yue Meng. Incorporating predictions in online graph coloring algorithms. *Discrete Applied Mathematics*, 379:434–445, 2026.
- [2] Joan Boyar, Leah Epstein, Lene Favrholdt, and Asaf Levin. Batch coloring of graphs. *Algorithmica*, 80, 10 2017.
- [3] András Gyárfás and Jenő Lehel. On-line and first fit colorings of graphs. *Journal of Graph Theory*, 12:217–227, 1988.
- [4] Magnús M. Halldórsson and Mario Szegedy. Lower bounds for on-line graph coloring. *Theoretical Computer Science*, 130(1):163–174, 1994.
- [5] Sandy Irani. Coloring inductive graphs on-line. *Algorithmica*, 11(1):53–72, 1994.
- [6] Yue Li, V. V. Narayan, and Denis Pankratov. Online coloring and a new type of adversary for online graph problems. *Algorithmica*, 84(5):1232–1251, 2022.
- [7] László Lovász, Michael Saks, and William T. Trotter. An on-line graph coloring algorithm with sublinear performance ratio. *Discrete Mathematics*, 75(1–3):319–325, 1989.