UNIVERSITÀ
DEGLI STUDI
FIRENZE

University of Florence

Year: 2025

Artificial Intelligence Project

Comparison between Min-Conflicts and Constraint Weighting Algorithms for the Map Coloring Problem

---

Federico Donati

Index

1 Introduction **1**

2 Min-Conflicts Algorithm **1**

3 Constraint Weighting Algorithm **2**

4 Conclusions **3**

1 Introduction

In this report, we analyze two algorithms designed to solve CSPs (Constraint Satisfaction Problems): the Min-Conflicts algorithm and the Constraint Weighting algorithm. We evaluate their performance on the classical map coloring problem.
The map is modeled as a connected graph, where nodes represent regions and edges represent borders between them. Each region must be assigned one of three possible colors such that no adjacent regions share the same color.
The experiments were conducted on a MacBook Air with an M2 chip.

This report not only presents raw performance data but also includes an in-depth analysis of each algorithm's behavior, success rate, and suitability under different conditions. The work is part of a broader investigation into heuristic-based techniques in CSPs, particularly those leveraging local search and constraint weighting dynamics.

2 Min-Conflicts Algorithm

The Min-Conflicts algorithm begins by generating an initial solution through random assignment of colors to variables. At each iteration (up to a predefined maximum number of steps), a randomly selected variable involved in a conflict is reassigned a value from its domain that minimizes the number of conflicts.
The implementation of the algorithm and its testing scripts can be found in the Codeberg project repository.
The experiment was executed for values of n ranging from 50 to 1000, in increments of 50. The algorithm was allowed a maximum of 100 steps per run.

Performance Results

The results obtained from the experiment are summarized in the following table:

Table 1: Performance of the Min-Conflicts Algorithm for Various Values of n

Input n	**Time (s)**	**Final Conflicts**
50	0.0004	0
100	0.0013	0
150	0.0024	0
200	0.0047	0
250	0.0076	0
300	0.0090	0
350	0.0129	0
400	0.0151	13
450	0.0174	29
500	0.0210	61
550	0.0217	83
600	0.0234	76
650	0.0257	98
700	0.0277	104
750	0.0295	112
800	0.0319	145

Continued on next page

Input n	Time (s)	Final Conflicts
850	0.0343	168
900	0.0360	177
950	0.0381	190
1000	0.0406	206

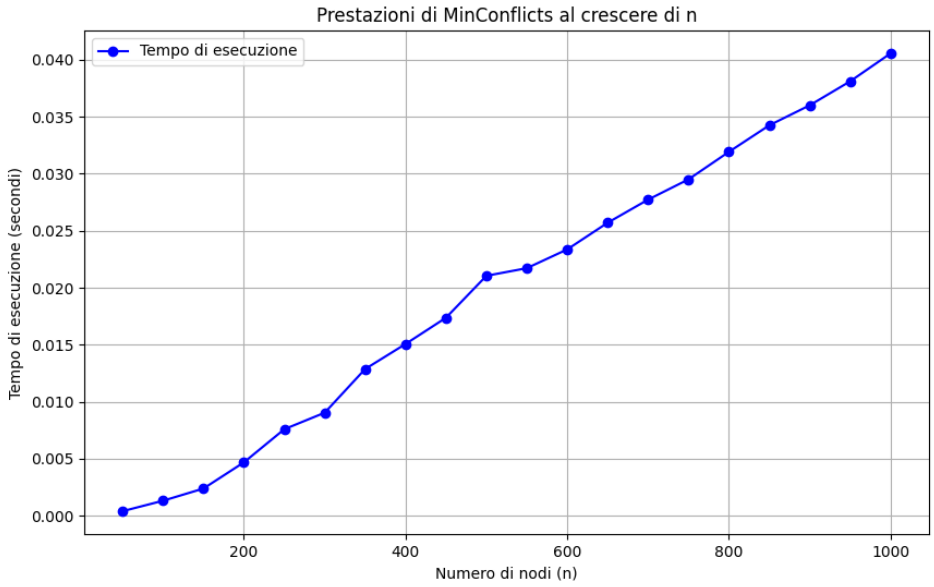

Figure 1: Performance Graph for the Min-Conflicts Algorithm

3 Constraint Weighting Algorithm

The Constraint Weighting algorithm also begins with a random assignment of colors to variables. At each iteration (up to a fixed maximum), it selects the edge (i.e., constraint) with the highest weight—if there are multiple, it selects one at random—and modifies one of the involved variables to a value from its domain that minimizes the total sum of constraint weights.
The implementation of this algorithm and its related testing routines can be found in the same Codeberg repository.
The experimental settings were identical to those of Min-Conflicts: n from 50 to 1000, with increments of 50, and a maximum of 100 steps.

Performance Results

Table 2: Performance of the Constraint Weighting Algorithm for Various Values of n

Input n	Time (s)	Final Conflicts
50	0.0003	0
100	0.0008	0
150	0.0018	0
200	0.0033	0
250	0.0042	0

Continued on next page

Input n	Time (s)	Final Conflicts
300	0.0067	0
350	0.0097	12
400	0.0107	0
450	0.0121	6
500	0.0139	50
550	0.0151	42
600	0.0173	80
650	0.0181	74
700	0.0200	115
750	0.0208	102
800	0.0223	147
850	0.0237	138
900	0.0253	175
950	0.0267	177
1000	0.0276	173

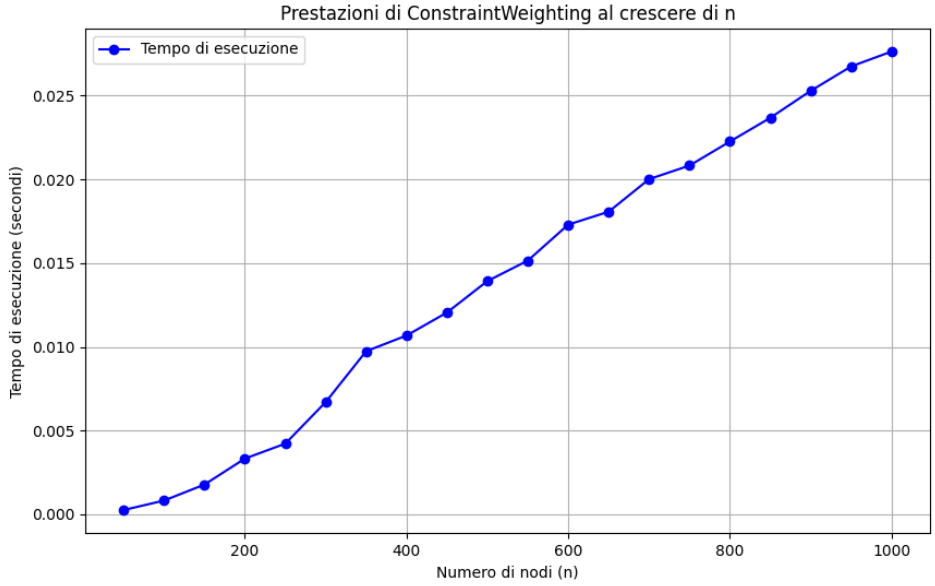

Figure 2: Performance Graph for the Constraint Weighting Algorithm

4 Conclusions

In terms of time efficiency, no significant difference emerged from the experiments. Nevertheless, Constraint Weighting slightly outperformed Min-Conflicts in execution speed.
However, when evaluating correctness—i.e., the ability to return a fully consistent solution—the Constraint Weighting algorithm showed its first failure at $n = 300$, whereas Min-Conflicts failed only starting from $n = 350$. The total number of successful runs (no conflicts) out of 20 was identical for both algorithms (7 successful completions each).
Despite both algorithms completing in negligible time, the maximum value for n had to be limited due to the considerable computational cost of generating the map's graph structure.

This observation suggests that further optimization of the map generation process would be essential for scaling these experiments to higher dimensions.

Reflections and Further Research

This comparative study highlights the practical trade-offs between two heuristic CSP-solving approaches: Min-Conflicts favors simplicity and is generally robust in sparse conflict scenarios, while Constraint Weighting adapts dynamically to problem structure through its weight-adjusting mechanism.
Potential directions for extending this work include:

- ◇ Implementing adaptive heuristics combining the strengths of both algorithms.
- ◇ Experimenting with additional constraint types or more complex map structures (e.g., real-world geography).
- ◇ Analyzing convergence behavior and stability across multiple random seeds to assess statistical consistency.
- ◇ Incorporating visualization of dynamic conflict resolution to better understand local search landscapes.