An Approximate Solution to the Minimum Vertex Cover Problem: The Hvala Algorithm

Frank Vega *

Posted Date: 28 April 2026

doi: 10.20944/preprints202506.0875.v13

Keywords: vertex cover; approximation algorithm; linear-time algorithm; ensemble heuristic; graph optimization; hardness of approximation

 doi:10.20944/preprints202506.0875.v13

Article

An Approximate Solution to the Minimum Vertex Cover Problem: The Hvala Algorithm

Frank Vega

Information Physics Institute, 840 W 67th St, Hialeah, FL 33012, USA; vega.frank@gmail.com

Abstract

We present the **Hvala** algorithm, a linear-time ensemble approximation method for the Minimum Vertex Cover problem. Hvala combines three complementary heuristics — a maximal-matching 2-approximation, a linear-time maximum-degree greedy implemented via a bucket-queue, and the degree-1 weighted-reduction "Hallelujah heuristic" studied in a companion work — with a redundant-vertex pruning post-processing step, and returns the smallest of the four resulting covers. **Theoretical guarantees.** We prove rigorously that Hvala achieves worst-case approximation ratio $\rho \leq 2$ for every finite, simple, undirected graph: the classical maximal-matching component alone already yields this bound, and the pruning step is shown to preserve cover validity while never increasing cover size. The companion work moreover establishes the strict pointwise inequality $|C_3| < 2 \cdot \mathrm{OPT}(G)$ on every finite simple graph — the Hallelujah heuristic's approximation ratio is asymptotic to 2 (strictly less than 2 on each graph, with supremum equal to 2 over all graphs) — and we show that this strict pointwise inequality is inherited by Hvala. Hvala runs in $\mathcal{O}(n+m)$ time and $\mathcal{O}(n+m)$ space. **Empirical performance.** We validate Hvala on two independent experimental studies totalling 239 instances. The first uses 109 vertex-cover instances of the public NPBench collection (41 FRB hard instances and 68 DIMACS clique-complement graphs, both with known optima), completed in 126.97 seconds: Hvala attains mean approximation ratio 1.021, with maximum 1.192 on a single Sanchis adversarial instance. The second evaluates Hvala on 130 real-world large graphs from the Network Data Repository (Cai's undirected simple graph collection), reaching up to 3 million vertices and 15 million edges, completed in approximately 95.5 minutes of cumulative solve time; on the 51 instances with published best-known cover sizes, mean ratio is 1.006 and maximum 1.036. **Prospects for a $\sqrt{2}-\epsilon$ bound.** Across the combined 160 instances with known optima, every approximation ratio lies below 1.414; 93.8% lie below 1.05 and 96.9% below 1.10. The natural open problem we propose as the continuation of this work is whether there exists a *fixed* constant $\epsilon > 0$ such that Hvala achieves uniform ratio $\sqrt{2}-\epsilon$ — either on all graphs (which, by SETH-based hardness, would imply P = NP) or, more realistically, on broad but restricted graph classes (bounded degree, bounded clique number, bounded treewidth, or structural families such as power-law and expander-like graphs). We do not prove such a bound here and do not claim one holds on all graphs; what we claim is that the combination of rigorous ≤ 2 guarantee, pointwise strict < 2 inequality, linear time, and observed ratios uniformly below 1.414 makes Hvala a plausible vehicle for such a refined analysis. The algorithm is publicly available via PyPI as the `hvala` package.

Keywords: vertex cover; approximation algorithm; linear-time algorithm; ensemble heuristic; graph optimization; hardness of approximation

MSC: 05C69; 68Q25; 90C27; 68W25

1. Introduction

The MINIMUM VERTEX COVER problem asks, for an undirected graph $G = (V, E)$, for the smallest subset $S \subseteq V$ such that every edge of G has at least one endpoint in S. It is one of Karp's original

 doi:10.20944/preprints202506.0875.v13

21 NP-complete problems [1] and underlies applications in wireless-network design, computational biology, scheduling, and VLSI.

Because exact minimum vertex covers cannot be computed in polynomial time unless P = NP, the problem has driven decades of work on approximation algorithms. The classical 2-approximation obtained by taking both endpoints of every edge of a maximal matching is folklore [2]; LP-based refinements by Karakostas [3] and Karpinski and Zelikovsky [4] reach factor $2 - \Theta(1/\sqrt{\log n})$, which is $2 - o(1)$ but does not match a constant $2 - \epsilon$. From the hardness side, Dinur and Safra [5] ruled out ratio below 1.3606 under P $\neq$ NP; Khot, Minzer and Safra [6–8] strengthened this to $\sqrt{2} - \epsilon$ under the Strong Exponential Time Hypothesis (SETH); and, under the Unique Games Conjecture [9], no constant factor below $2 - \epsilon$ is achievable [10]. A polynomial-time algorithm with constant ratio $\rho < \sqrt{2}$ would therefore resolve P versus NP, and already an unconditional $2 - \epsilon$ constant is considered beyond reach of current techniques.

Scope and contribution. Against this backdrop, this paper is deliberately modest in its theoretical claims and stays within rigorously provable territory. The contributions are:

1. A linear-time ensemble algorithm (**Hvala**, Algorithm 1) that wraps three complementary linear-time heuristics — (i) a maximal-matching 2-approximation, (ii) a bucket-queue max-degree greedy, and (iii) the Hallelujah degree-1 weighted-reduction heuristic [11] — inside a redundant-vertex pruning step, and returns the smallest resulting cover.
2. A *rigorous proof* (Theorem 2) that Hvala achieves worst-case approximation ratio $\rho \leq 2$ on every finite simple graph. The proof hinges on the maximal-matching component and is self-contained.
3. A *strict pointwise inequality* $|S| < 2 \cdot \mathrm{OPT}(G)$ on every finite simple graph (Corollary 1), inherited from the companion paper [11]. The Hallelujah heuristic's approximation ratio is asymptotic to 2 — strictly less than 2 on each graph, with supremum equal to 2 — so no constant strictly smaller than 2 bounds it uniformly; but the pointwise strict inequality on each graph is preserved by the minimum-selection and pruning steps of Hvala.
4. An empirical evaluation on two independent experimental studies totalling 239 instances: 109 structured hard instances from the NPBench benchmark collection [12] (41 FRB hard random graphs and 68 DIMACS clique-complement graphs, all with known optima) and 130 real-world large graphs from the Network Data Repository [13] (biological, social, collaboration, web, infrastructure, and scientific-computing networks, reaching up to 2,523,386 vertices and 15,245,729 edges), reporting solution quality, running time, and a breakdown by graph family.

The remainder of the paper is organised as follows. Section 3 describes the Hvala algorithm in detail. Section 4 establishes the linear-time complexity. Section 5 contains the approximation-ratio analysis (rigorous ≤ 2 bound and the strict pointwise < 2 inheritance). Section 6 reports two experimental studies: the NPBench structured-hard-instance benchmark (Section 6.1) and a real-world large-graph benchmark drawn from the Network Data Repository (Section 6.2). Section 7 discusses the empirical–theoretical gap, hardness barriers, and the prospects of Hvala as a candidate for refined analysis below the $\sqrt{2}$ threshold on restricted graph classes (Section 7.3); Section 8 concludes.

2. Research Data and Implementation

To facilitate reproducibility and community adoption, we developed the open-source Python package HVALA: *Approximate Vertex Cover Solver*, available via the Python Package Index (PyPI) [14]. This implementation encapsulates the full ensemble algorithm — including the maximal-matching 2-approximation, the bucket-queue max-degree greedy, the Hallelujah degree-1 weighted-reduction subroutine, and the redundant-vertex pruning post-processing step — while guaranteeing an approximation ratio at most 2 (with pointwise strict inequality < 2 on every finite simple graph) through rigorous validation. The package integrates seamlessly with NetworkX for graph handling and supports both unweighted and weighted instances. Code metadata, including versioning, licensing, and dependencies, is detailed in Table 1.

 doi:10.20944/preprints202506.0875.v13

Table 1. Code metadata for the HVALA package.

Nr.	Code metadata description	Metadata
C1	Current code version	v0.1.0
C2	Permanent link to code/repository used for this code version	https://github.com/frankvegadelgado/hvala
C3	Permanent link to Reproducible Capsule	https://pypi.org/project/hvala/
C4	Legal Code License	MIT License
C5	Code versioning system used	git
C6	Software code languages, tools, and services used	Python
C7	Compilation requirements, operating environments & dependencies	Python $\geq$ 3.12, NetworkX $\geq$ 3.4.2

3. The Hvala Algorithm

3.1. Overview

Given a simple undirected graph $G = (V, E)$, Hvala first performs trivial preprocessing (remove self-loops and isolated vertices) and then computes four candidate vertex covers:

C_1 — **Maximal-matching cover.** Compute a maximal matching M of G and let $C_1 = \bigcup_{(u,v)\in M}\{u, v\}$. This is the classical 2-approximation of [2].

C_2 — **Bucket-queue max-degree greedy.** Repeatedly select a vertex of maximum current degree into the cover, removing it and its incident edges, until no edges remain. Implemented in linear total time using a bucket queue indexed by degree.

C_3 — **Hallelujah degree-1 reduction.** Build an auxiliary graph G' by splitting every vertex u of degree k into k auxiliary copies $(u, 0), \ldots, (u, k-1)$, each connected to exactly one of u's neighbours, and assigning weight $1/k$ to every such auxiliary vertex. G' has maximum degree at most 1 on the auxiliary side, so a minimum weighted vertex cover on G' is obtained by picking, per edge of G', the endpoint of smaller weight (with lexicographic tie-breaking). Projecting the selected auxiliary vertices (u, i) back to their original u yields a valid cover of G [11].

$\tilde{C}_4$ — **Pruned union.** Start from $C_1 \cup C_2 \cup C_3$ and apply redundant-vertex pruning (Algorithm 6) *once*, directly yielding the fourth candidate $\tilde{C}_4$.

The candidates C_1, C_2, C_3 are then *each* individually passed through redundant-vertex pruning to obtain $\tilde{C}_1, \tilde{C}_2, \tilde{C}_3$, and the algorithm returns the smallest among $\tilde{C}_1, \tilde{C}_2, \tilde{C}_3, \tilde{C}_4$. Note that C_1 is included as a *worst-case safety net*: its value is guaranteed to be at most $2 \cdot \mathrm{OPT}$, and since the algorithm returns $\min(|\tilde{C}_1|, |\tilde{C}_2|, |\tilde{C}_3|, |\tilde{C}_4|)$, this guarantee propagates to the final output regardless of how C_2, C_3, and the union behave.

 doi:10.20944/preprints202506.0875.v13

3.2. Main Algorithm

Algorithm 1: Hvala: FINDVERTEXCOVER(G)

Input: Simple undirected graph $G = (V, E)$
Output: Vertex cover $S \subseteq V$ satisfying $|S| \leq 2 \cdot \mathrm{OPT}(G)$

Remove self-loops and isolated vertices from G;
if $E(G) = \emptyset$ **then**
  return $\emptyset$;
end
Build adjacency table $\mathrm{adj}[v] = N_G(v)$ for every $v \in V$;
// Three base heuristics (all linear time)
$C_1 \leftarrow$ MAXIMALMATCHINGVC(G) ; // Algorithm 2
$C_2 \leftarrow$ BUCKETDEGREEGREEDY(adj) ; // Algorithm 3
$C_3 \leftarrow$ HALLELUJAHREDUCTION(G) ; // Algorithm 4; [11]
// Individual pruning of every candidate
foreach $i \in \{1, 2, 3\}$ **do**
  $\tilde{C}_i \leftarrow$ PRUNEREDUNDANT(adj, C_i);
end
// Pruned union
$\tilde{C}_4 \leftarrow$ PRUNEREDUNDANT(adj, $C_1 \cup C_2 \cup C_3$) ; // Algorithm 6
return $\arg\min_{i \in \{1,2,3,4\}} |\tilde{C}_i|$;

3.3. Subroutines

Algorithm 2: MAXIMALMATCHINGVC(G)

Input: Graph G
Output: Vertex cover C_1

$M \leftarrow$ a maximal matching of G;
$C_1 \leftarrow \emptyset$;
foreach $(u, v) \in M$ **do**
  $C_1 \leftarrow C_1 \cup \{u, v\}$;
end
return C_1;

 doi:10.20944/preprints202506.0875.v13

Algorithm 3: BUCKETDEGREEGREEDY(adj)

Input: Adjacency table adj of a graph $G = (V, E)$
Output: Vertex cover C_2

$\deg[v] \leftarrow |\text{adj}[v]|$ for all $v \in V$;
$\Delta \leftarrow \max_v \deg[v]$;
Create buckets $B[0], B[1], \ldots, B[\Delta]$, double-ended queues;
foreach $v \in V$ **do**
  append v to $B[\deg[v]]$;
end
$C_2 \leftarrow \emptyset$; *removed* $\leftarrow \emptyset$;
for $d = \Delta$ ***down to*** 1 **do**
  while $B[d] \neq \emptyset$ **do**
    pop v from the front of $B[d]$;
    if $v \in$ *removed* ***or*** $\deg[v] \neq d$ **then**
      continue;
    end
    $C_2 \leftarrow C_2 \cup \{v\}$; *removed* $\leftarrow$ *removed* $\cup \{v\}$;
    foreach $u \in \text{adj}[v] \setminus$ *removed* **do**
      $\deg[u] \leftarrow \deg[u] - 1$;
      append u to $B[\deg[u]]$;
    end
  end
end
return C_2;

Algorithm 4: HALLELUJAHREDUCTION(G) — the degree-1 weighted reduction of [11]

Input: Graph $G = (V, E)$
Output: Vertex cover C_3

Build an auxiliary graph $G' = (V', E')$::
foreach $u \in V$ *with degree* $k > 0$ *and neighbours* $v_0, v_1, \ldots, v_{k-1}$ **do**
  Add auxiliary vertex (u, i) and edge $\{(u, i), v_i\}$ to G' for every $i = 0, \ldots, k-1$;
  Set $w((u, i)) \leftarrow 1/k$;
end
$D_{\text{uw}} \leftarrow$ MINVCDEGREE1(G', uniform weights);
$D_{\text{w}} \leftarrow$ MINVCDEGREE1(G', w);
$S_{\text{uw}} \leftarrow \{u : (u, i) \in D_{\text{uw}} \text{ for some } i\} \cup (D_{\text{uw}} \cap V)$;
$S_{\text{w}} \leftarrow \{u : (u, i) \in D_{\text{w}} \text{ for some } i\} \cup (D_{\text{w}} \cap V)$;
return smaller of S_{uw} and S_{w};

 doi:10.20944/preprints202506.0875.v13

Algorithm 5: MINVCDEGREE1(G', w) — exact weighted VC on a max-degree-1 graph

Input: Graph G' of maximum degree 1; weight w
Output: Minimum weighted vertex cover D

$D \leftarrow \emptyset$; *visited* $\leftarrow \emptyset$;
foreach $v \in V(G')$ *with* $v \notin$ *visited* **do**
  if $\deg(v) = 1$ **then**
    $u \leftarrow$ unique neighbour of v;
    if $u \notin$ *visited* **then**
      if $w(v) < w(u)$ ***or*** $(w(v) = w(u)$ ***and*** $v < u)$ **then**
        $D \leftarrow D \cup \{v\}$;
      else
        $D \leftarrow D \cup \{u\}$;
      end
      visited $\leftarrow$ *visited* $\cup \{v, u\}$;
    end
  end
end
return D;

Algorithm 6: PRUNEREDUNDANT(adj, C) — linear-time redundant-vertex pruning

Input: Adjacency table adj of G and a vertex cover C of G
Output: A vertex cover $C' \subseteq C$ of G

$L \leftarrow$ a fixed list copy of C (iteration order is immaterial for the proof);
foreach $v \in L$ **do**
  if *every* $u \in$ adj$[v]$ *is currently in* C **then**
    $C \leftarrow C \setminus \{v\}$;
  end
end
return C;

4. Complexity Analysis

Theorem 1 (Linear-time and linear-space)**.** *Hvala runs in* $\mathcal{O}(n + m)$ *time and* $\mathcal{O}(n + m)$ *space, where* $n = |V|$ *and* $m = |E|$.

Proof. Preprocessing and construction of the adjacency table are $\mathcal{O}(n + m)$.

Maximal matching can be computed in $\mathcal{O}(n + m)$ by the greedy linear-time procedure that scans edges once and adds every edge both of whose endpoints are still unmatched. Building C_1 from M is $\mathcal{O}(|M|) \leq \mathcal{O}(m)$.

Bucket-queue max-degree greedy. Each vertex is inserted into a bucket at most once per decrement of its degree; across the whole execution the total number of bucket insertions is bounded by $\sum_v \deg(v) = 2m$, and each insertion/removal is $\mathcal{O}(1)$. The outer loop over d performs $\mathcal{O}(\Delta) \leq \mathcal{O}(n)$ constant-time bucket checks. Total time: $\mathcal{O}(n + m)$.

Hallelujah reduction. The auxiliary graph G' has exactly $2m$ vertices and m edges (one edge per edge of G, with auxiliary vertices added on both endpoints when both are of positive degree). MINVCDEGREE1 visits every vertex once and its single neighbour once, hence runs in $\mathcal{O}(|V(G')| + |E(G')|) = \mathcal{O}(n + m)$.

Pruning. For each $v \in C$, checking whether all neighbours are in C is $\mathcal{O}(\deg(v))$; summed over all $v \in C \subseteq V$ the total work is at most $\sum_{v \in V} \deg(v) = 2m$. Each pruning call is therefore $\mathcal{O}(n + m)$, and the algorithm performs a constant number of pruning calls.

Space is dominated by the adjacency table and the auxiliary graph G', both $\mathcal{O}(n + m)$. □

 doi:10.20944/preprints202506.0875.v13

5. Approximation Ratio Analysis

We now establish the worst-case approximation guarantees of Hvala, in two stages. First, a self-contained proof that Hvala always returns a cover of size at most $2 \cdot \mathrm{OPT}$ (Theorem 2); this is the baseline guarantee. Second, an inheritance argument (Corollary 1) showing that Hvala satisfies the strict pointwise inequality $|S| < 2 \cdot \mathrm{OPT}(G)$ on every finite simple graph G, mirroring the analogous property proved for the Hallelujah heuristic in the companion paper [11]. Both statements are needed: Theorem 2 gives the absolute ≤ 2 bound, while Corollary 1 records that the inequality is in fact strict on each graph, even though — as explained in Section 5.3 below — the supremum of the ratio over all graphs still equals 2.

Throughout this section, $G = (V, E)$ is a finite simple undirected graph without self-loops, and $\mathrm{OPT}(G)$ denotes the size of a minimum vertex cover of G. Isolated vertices contribute 0 to OPT, so removing them (as the algorithm does in preprocessing) leaves OPT unchanged.

5.1. A Lemma about Redundant-Vertex Pruning

Lemma 1 (Pruning preserves validity and never increases size)**.** *Let C be a vertex cover of G and let $C' = \textsc{PruneRedundant}(\mathrm{adj}, C)$. Then $C' \subseteq C$ and C' is also a vertex cover of G.*

Proof. That $C' \subseteq C$ is clear from the procedure (it only ever removes elements).

We prove by induction on the iteration count that the invariant "C is a vertex cover of G" holds throughout PRUNEREDUNDANT.

Base case. At the start, C is a vertex cover of G by hypothesis.

Inductive step. Suppose the invariant holds just before iteration i, at which we are considering vertex $v \in L$. Two cases.

Case 1: v is not removed at iteration i. Then C is unchanged, and the invariant is preserved trivially.

Case 2: v is removed at iteration i. The removal condition is that every neighbour u of v in G is currently in C. Consider any edge $e = (x, y) \in E$:

- If e is not incident to v, neither of its endpoints is touched; the inductive hypothesis says some endpoint of e was in C before iteration i, and both endpoints remain unaffected, so the property survives.
- If $e = (v, u)$ is incident to v, then by the removal condition, u is in C just before v is removed, and u is not the vertex being removed, so u remains in C after the removal. Hence e is still covered by u.

Thus the invariant is preserved.

Since the loop terminates after a finite number of iterations, the invariant holds at the end, and C' is a vertex cover of G. □

It is instructive, though not logically necessary for what follows, to note the following strengthening: once a vertex v is removed, no neighbour of v can subsequently be removed, because the "all neighbours currently in C" test would fail (with v itself missing from C). In particular, after PRUNEREDUNDANT, for every edge (v, u), at most one of v, u has been removed. This reinforces Lemma 1.

5.2. The Rigorous $\rho \leq 2$ Bound

Theorem 2 (Worst-case 2-approximation)**.** *For every finite simple undirected graph G, the output S of $\textsc{FindVertexCover}(G)$ (Algorithm 1) is a vertex cover of G satisfying*

$$|S| \;\leq\; 2 \cdot \mathrm{OPT}(G).$$

Proof. Let G_0 be the graph after preprocessing (self-loops and isolated vertices removed). As noted above, $\mathrm{OPT}(G_0) = \mathrm{OPT}(G)$, and any vertex cover of G_0 is a vertex cover of G. We work with G_0 in what follows.

 doi:10.20944/preprints202506.0875.v13

Step 1: C_1 is a vertex cover of G_0 of size at most $2 \cdot \mathrm{OPT}(G_0)$.

Let M be the maximal matching computed in Algorithm 2. Since M is maximal, every edge $e \in E(G_0)$ shares a vertex with some edge of M — otherwise $M \cup \{e\}$ would be a larger matching, contradicting maximality. Therefore, at least one endpoint of e lies in $C_1 = \bigcup_{(u,v)\in M}\{u, v\}$, i.e. C_1 is a vertex cover of G_0.

Let C^* be any minimum vertex cover of G_0, so $|C^*| = \mathrm{OPT}(G_0)$. Since the edges of M are pairwise vertex-disjoint, and each of these edges must be covered by C^*, distinct edges of M contribute distinct vertices to C^* (one endpoint each). Hence $|C^*| \geq |M|$. Consequently,

$$|C_1| \;=\; 2|M| \;\leq\; 2|C^*| \;=\; 2 \cdot \mathrm{OPT}(G_0).$$

Step 2: Pruning C_1 does not increase its size.

By Lemma 1 applied to C_1, the pruned $\tilde{C}_1 = \textsc{PruneRedundant}(\mathrm{adj}, C_1)$ is still a vertex cover of G_0, and $\tilde{C}_1 \subseteq C_1$ implies $|\tilde{C}_1| \leq |C_1|$. Combining with Step 1:

$$|\tilde{C}_1| \;\leq\; |C_1| \;\leq\; 2 \cdot \mathrm{OPT}(G_0).$$

Step 3: The output S satisfies $|S| \leq |\tilde{C}_1|$.

By construction, the algorithm returns $S = \arg\min_{i\in\{1,2,3,4\}} |\tilde{C}_i|$. Hence $|S| \leq |\tilde{C}_1|$, provided $\tilde{C}_1$ is a vertex cover so that the minimum is well-defined over valid covers — and it is, by Step 2. (Note that $\tilde{C}_2, \tilde{C}_3, \tilde{C}_4$ are also valid covers: $\tilde{C}_2, \tilde{C}_3$ by Lemma 1 applied to C_2, C_3, which are valid covers since C_2 is produced by a process that terminates only when all edges are covered and C_3 is the standard vertex-cover projection of the reduction of [11]; and $\tilde{C}_4$ by Lemma 1 applied to the union $C_1 \cup C_2 \cup C_3$, which is a valid cover as a superset of the valid cover C_1.)

Combining Steps 1–3, $|S| \leq 2 \cdot \mathrm{OPT}(G_0) = 2 \cdot \mathrm{OPT}(G)$. □

The constant 2 in Theorem 2 is a *uniform worst-case* bound on $|S|/\mathrm{OPT}(G)$ that holds over all finite simple graphs. Achieving a strictly smaller *uniform constant* $2 - \epsilon$ with a simple combinatorial algorithm is a well-known open problem, because an unconditional constant $2 - \epsilon$ would improve over the best known $2 - \Theta(1/\sqrt{\log n})$ [3] and is UGC-hard [10]. We do not claim such an improvement here. What we do obtain, from the companion paper, is a weaker but non-trivial statement: the inequality $|S| \leq 2 \cdot \mathrm{OPT}(G)$ is in fact *strict* on every particular graph.

5.3. Inheritance of the Pointwise Strict Inequality from Hallelujah

The companion paper [11] establishes the following property of the degree-1 weighted-reduction heuristic (our C_3): for every finite simple undirected graph G, the cover $C_3 = \textsc{HallelujahReduction}(G)$ satisfies

$$|C_3| \;<\; 2 \cdot \mathrm{OPT}(G).$$

The inequality is strict on each graph. At the same time, the supremum of $|C_3|/\mathrm{OPT}(G)$ over all finite simple graphs equals 2: the Hallelujah ratio is *asymptotic* to 2, that is, for every $\epsilon > 0$ there exists a graph G_ϵ on which $|C_3|/\mathrm{OPT}(G_\epsilon) > 2 - \epsilon$. Consequently, there is no single constant strictly less than 2 that uniformly bounds the ratio over all graphs. We refer the reader to [11] for the full proof of both facts and use only the pointwise strict inequality below.

Corollary 1 (Strict pointwise inequality for Hvala)**.** *For every finite simple undirected graph G, the output S of Algorithm 1 satisfies*

$$|S| \;<\; 2 \cdot \mathrm{OPT}(G).$$

The supremum of $|S|/\mathrm{OPT}(G)$ over all finite simple graphs is equal to 2: no uniform constant strictly less than 2 bounds the Hvala ratio either.

 doi:10.20944/preprints202506.0875.v13

Proof. *Strict pointwise inequality.* Let $\tilde{C}_3 = \textsc{PruneRedundant}(\text{adj}, C_3)$. By Lemma 1, $\tilde{C}_3$ is a vertex cover of G with $|\tilde{C}_3| \leq |C_3|$. By the Hallelujah property quoted above, $|C_3| < 2 \cdot \text{OPT}(G)$. Hence $|\tilde{C}_3| < 2 \cdot \text{OPT}(G)$. Since $S = \arg\min_{i \in \{1,2,3,4\}} |\tilde{C}_i|$, we have $|S| \leq |\tilde{C}_3| < 2 \cdot \text{OPT}(G)$.

Supremum equals 2. The upper bound $\sup_G |S|/\text{OPT}(G) \leq 2$ is immediate from Theorem 2. For the matching lower bound, consider any graph G_ϵ on which $|C_3|/\text{OPT}(G_\epsilon) > 2 - \epsilon$ (such G_ϵ exist by the asymptotic-to-2 property of Hallelujah). If on such a family the four Hvala candidates $\tilde{C}_1, \tilde{C}_2, \tilde{C}_3, \tilde{C}_4$ all have ratio approaching 2, then so does $|S|/\text{OPT}(G_\epsilon)$. Since the property of ratio tending to 2 cannot be ruled out for $|S|$ without ruling it out for each candidate — and in particular a uniform constant $2 - \delta$ bounding $|S|$ would contradict the absence of such a constant for Hallelujah alone on the very families where Hallelujah is the tightest candidate — we conclude that $\sup_G |S|/\text{OPT}(G) = 2$. □

Two remarks clarify the interplay between Theorem 2 and Corollary 1.

First, Theorem 2 is *not* rendered redundant by Corollary 1. The theorem gives a uniform, self-contained, non-strict ≤ 2 bound whose proof does not depend on [11] at all. The corollary strengthens this to a strict inequality on each particular graph but relies on the companion paper's analysis of the Hallelujah reduction. Readers who wish to audit Hvala against only the simplest assumptions thus have the ≤ 2 guarantee with a proof entirely contained here.

Second, the strict inequality in Corollary 1 is pointwise only: it does not provide a uniform constant $2 - \epsilon$, and no such constant is known for either the Hallelujah component or Hvala. Establishing a uniform $2 - \epsilon$ constant for any simple polynomial-time algorithm would improve over [3] and, under the Unique Games Conjecture, is not possible [9,10]. The statement "the ratio tends to 2" should be read in this precise sense: on every finite graph the ratio is strictly below 2, but by choosing progressively harder graphs one can make it arbitrarily close to 2.

5.4. Other Candidates

We have used C_1 and C_3 in the proofs; the roles of C_2 and $\tilde{C}_4$ are complementary rather than load-bearing:

- C_2 (bucket-queue max-degree greedy) has no general worst-case ratio better than $\Theta(\log \Delta)$ (Johnson's classical bound), but it is very strong on near-regular and clique-like graphs and is included because, on those families, it is frequently optimal or near-optimal. Its presence in the minimum cannot worsen the bound.
- $\tilde{C}_4$ is the pruning of the union $C_1 \cup C_2 \cup C_3$. Because $|\tilde{C}_4| \leq |C_1 \cup C_2 \cup C_3|$ may be larger or smaller than each individual $|\tilde{C}_i|$ for $i \in \{1,2,3\}$, its role is best understood as occasionally exploiting structural overlaps between the three base heuristics that per-candidate pruning alone cannot resolve.

Neither C_2 nor $\tilde{C}_4$ is required for the bounds of Theorem 2 or Corollary 1.

6. Experimental Validation

We evaluate Hvala on two independent experimental studies totalling 239 instances. Section 6.1 reports results on 109 structured hard instances of the public NPBench collection [12] with known optima, and Section 6.2 reports results on 130 real-world large graphs drawn from the Network Data Repository [13], reaching millions of vertices. Both studies use the same implementation of Algorithm 1 on commodity hardware (Intel Core i7-1165G7 at 2.80 GHz, 32 GB RAM, single-threaded Python 3.12 with NetworkX 3.4.2).

6.1. Experiment 1: Structured Hard Instances (NPBench)

6.1.1. Setup

We evaluate Hvala on 109 vertex-cover instances of the public NPBench collection [12], comprising two families for which the optimum (or a tight best-known value) is publicly available:

 doi:10.20944/preprints202506.0875.v13

1. **41 FRB hard instances** (from NPBench Section "Vertex Cover instances", originally from Ke Xu's benchmark repository), with known minimum vertex cover sizes ranging from 420 to 3900.
2. **68 DIMACS clique-complement instances** (from NPBench Section "Clique complement graphs"), constructed as the complements of the DIMACS Second Implementation Challenge maximum-clique instances. The optimum vertex cover of the complement equals $n - \omega(G)$, where $\omega(G)$ is the maximum clique size of the original graph; we use the maximum-clique values compiled on Mascia's DIMACS benchmark page [15]. For the two instances `C500.9` and `C1000.9` the clique number is only known to be a best-known lower bound ($\omega \geq 57$ and $\omega \geq 68$ respectively), so the values shown are the best-known upper bounds on OPT and the reported ratio is itself a lower bound on the true ratio (marked with †).

Cumulative solve time over all 109 instances is 126.97 seconds.

6.1.2. Results

Table 2 reports, for every FRB instance, the known optimum, the cover size produced by Hvala, the wall-clock solve time, and the approximation ratio. Tables 3 and 4 report the same quantities for the DIMACS clique-complement instances, split alphabetically into two halves so that each fits on a standard page.

Table 2. Hvala on the 41 FRB vertex-cover instances of NPBench [12].

Instance	Known OPT	Hvala size	Time	Ratio
frb30-15-1	420	428	214.1ms	1.019
frb30-15-2	420	429	219.7ms	1.021
frb30-15-3	420	427	206.9ms	1.017
frb30-15-4	420	429	1.45s	1.021
frb30-15-5	420	427	249.9ms	1.017
frb35-17-1	560	569	403.9ms	1.016
frb35-17-2	560	570	443.8ms	1.018
frb35-17-3	560	568	455.1ms	1.014
frb35-17-4	560	570	445.5ms	1.018
frb35-17-5	560	568	484.9ms	1.014
frb40-19-1	720	730	716.4ms	1.014
frb40-19-2	720	730	705.6ms	1.014
frb40-19-3	720	731	725.1ms	1.015
frb40-19-4	720	732	756.8ms	1.017
frb40-19-5	720	730	759.7ms	1.014
frb45-21-1	900	912	1.07s	1.013
frb45-21-2	900	911	1.19s	1.012
frb45-21-3	900	912	1.16s	1.013
frb45-21-4	900	912	1.10s	1.013
frb45-21-5	900	912	1.18s	1.013
frb50-23-1	1100	1111	1.57s	1.010
frb50-23-2	1100	1113	1.71s	1.012
frb50-23-3	1100	1117	1.74s	1.015
frb50-23-4	1100	1113	1.65s	1.012
frb50-23-5	1100	1112	1.69s	1.011
frb53-24-1	1219	1235	1.95s	1.013
frb53-24-2	1219	1234	2.09s	1.012
frb53-24-3	1219	1235	2.09s	1.013
frb53-24-4	1219	1232	2.14s	1.011
frb53-24-5	1219	1235	2.00s	1.013
frb56-25-1	1344	1358	2.48s	1.010
frb56-25-2	1344	1358	2.44s	1.010
frb56-25-3	1344	1359	2.33s	1.011
frb56-25-4	1344	1358	2.46s	1.010
frb56-25-5	1344	1361	2.54s	1.013
frb59-26-1	1475	1492	2.81s	1.012
frb59-26-2	1475	1492	2.98s	1.012
frb59-26-3	1475	1494	4.15s	1.013
frb59-26-4	1475	1493	4.38s	1.012
frb59-26-5	1475	1491	4.56s	1.011
frb100-40	3900	3931	16.82s	1.008

 doi:10.20944/preprints202506.0875.v13

Table 3. Hvala on the 68 DIMACS clique-complement instances of NPBench [12] — Part 1 of 2 (`brock`, `c-fat`, `C`, `gen`, `hamming`). †: best-known upper bound on OPT (maximum clique not confirmed optimal [15]); the ratio is then a lower bound.

Instance	Known OPT	Hvala size	Time	Ratio
brock200_1	179	183	59.3ms	1.022
brock200_2	188	192	134.8ms	1.021
brock200_3	185	189	101.5ms	1.022
brock200_4	183	187	65.6ms	1.022
brock400_1	373	381	253.4ms	1.021
brock400_2	371	379	262.8ms	1.022
brock400_3	369	381	254.5ms	1.033
brock400_4	367	378	259.6ms	1.030
brock800_1	777	785	2.12s	1.010
brock800_2	776	783	2.41s	1.009
brock800_3	775	784	2.30s	1.012
brock800_4	774	785	3.39s	1.014
c-fat200-1	188	188	450.9ms	1.000
c-fat200-2	176	176	809.6ms	1.000
c-fat200-5	142	142	266.4ms	1.000
c-fat500-1	486	486	2.19s	1.000
c-fat500-10	374	374	1.62s	1.000
c-fat500-2	474	474	2.22s	1.000
c-fat500-5	436	436	2.19s	1.000
C1000.9	932†	945	1.05s	1.014
C125.9	91	92	10.0ms	1.011
C250.9	206	214	31.9ms	1.039
C500.9	443†	454	276.9ms	1.025
gen200_p0.9_44	156	167	22.6ms	1.071
gen200_p0.9_55	145	163	22.3ms	1.124
gen400_p0.9_55	345	356	83.6ms	1.032
gen400_p0.9_65	335	359	129.0ms	1.072
gen400_p0.9_75	325	358	122.6ms	1.102
hamming10-2	512	512	60.9ms	1.000
hamming10-4	984	992	1.73s	1.008
hamming6-2	32	32	2.1ms	1.000
hamming6-4	60	60	21.0ms	1.000
hamming8-2	128	128	14.2ms	1.000
hamming8-4	240	240	129.3ms	1.000

 doi:10.20944/preprints202506.0875.v13

Table 4. Hvala on the 68 DIMACS clique-complement instances of NPBench [12] — Part 2 of 2 (`johnson`, `keller`, `MANN`, `p_hat`, `san`, `sanr`).

Instance	Known OPT	Hvala size	Time	Ratio
johnson16-2-4	112	112	18.2ms	1.000
johnson32-2-4	480	480	365.8ms	1.000
johnson8-2-4	24	24	2.2ms	1.000
johnson8-4-4	56	56	7.3ms	1.000
keller4	160	162	77.4ms	1.012
keller5	749	759	1.34s	1.013
MANN_a27	252	253	15.9ms	1.004
MANN_a45	690	695	27.1ms	1.007
MANN_a81	2221	2225	82.8ms	1.002
MANN_a9	29	29	0.0ms	1.000
p_hat1000-3	932	942	2.79s	1.011
p_hat300-1	292	292	751.1ms	1.000
p_hat300-2	275	277	362.0ms	1.007
p_hat300-3	264	268	170.7ms	1.015
p_hat500-1	491	492	1.61s	1.002
p_hat500-2	464	469	1.21s	1.011
p_hat500-3	450	454	560.2ms	1.009
p_hat700-1	689	693	3.78s	1.006
p_hat700-2	656	658	2.82s	1.003
p_hat700-3	638	642	1.31s	1.006
san200_0.7_1	170	184	59.4ms	1.082
san200_0.7_2	182	188	201.1ms	1.033
san200_0.9_1	130	155	21.2ms	1.192
san200_0.9_2	140	162	19.2ms	1.157
san200_0.9_3	156	169	26.3ms	1.083
san400_0.5_1	387	393	609.8ms	1.016
san400_0.7_1	360	379	443.1ms	1.053
san400_0.7_2	370	385	364.2ms	1.041
san400_0.7_3	378	388	365.2ms	1.026
san400_0.9_1	300	348	86.3ms	1.160
sanr200_0.7	182	184	124.4ms	1.011
sanr200_0.9	158	160	21.7ms	1.013
sanr400_0.5	387	388	718.1ms	1.003
sanr400_0.7	379	383	990.0ms	1.011

6.1.3. Summary Statistics

Of the 109 instances with a known optimum (or best-known bound), Hvala achieves:

- **Mean approximation ratio:** 1.021 (FRB block: 1.014; DIMACS clique-complement block: 1.025).
- **Exact optimality:** 18 instances solved with ratio 1.000, concentrated in the `c-fat`, `hamming`, `johnson`, `MANN_a9`, and `p_hat300-1` families.
- **Maximum ratio observed:** 1.192 on `san200_0.9_1` (a Sanchis instance constructed with an embedded clique of size 70). The five worst ratios are all on Sanchis `san/gen` adversarial instances, which are specifically engineered to hide large cliques; on these dense, small, carefully constructed graphs, ensemble heuristics are known to degrade relative to specialised exact solvers.
- **Runtime:** total cumulative solve time across all 109 instances is 126.97 seconds (80.53 s on FRB + 46.44 s on DIMACS). Per-instance times range from under 10 ms (smallest graphs) to 16.82 s (`frb100-40`, the largest FRB instance).

Every single observed ratio is strictly below 2, consistent with Theorem 2 and Corollary 1. The consistent empirical proximity to OPT, especially on the combinatorially-structured DIMACS complements, suggests that in practice Hvala operates far below its proven worst-case bound.

6.2. Experiment 2: Real-World Large Graphs

6.2.1. Setup

This section presents comprehensive experimental results of the Hvala algorithm on real-world large graphs from the Network Data Repository [13]. The benchmark suite consists of 130 instances from the complete collection of 139 undirected simple largest graphs distributed by Cai [13]. Nine instances are excluded: three graphs (`ca-hollywood-2009`, `socfb-uci-uni`, `soc-orkut`) ex-

 doi:10.20944/preprints202506.0875.v13

ceed the 32 GB RAM limit of our test hardware when loaded through NetworkX, and six further graphs (`inf-road-usa`, `sc-ldoor`, `soc-livejournal`, `soc-pokec`, `socfb-A-anon`, `socfb-B-anon`) were dropped to keep the experiment tractable within a single session; these nine exclusions represent the most memory-intensive instances in the collection. The retained 130 instances span biological networks, scientific collaboration graphs, email networks, social networks (including Facebook), infrastructure (power grids, routers, autonomous systems, road networks), web graphs, retweet networks, strongly connected components, and scientific computing networks (FEM and structural problems). Graphs range from 2 vertices (`scc_rt_http`) to 2,523,386 vertices and 15,245,729 edges (the largest by edges is `ca-coauthors-dblp`).

Because the Network Data Repository does not provide certified minimum vertex cover values for most of these instances, we rely on the *best-known approximate optimum* values compiled by the Milagro experiment [16] on the same collection. For 51 of the 130 instances such a reference value is available (of which 29 are certified optima on tree-like components); for the remaining 79 instances we list "Unknown".

Every returned cover satisfies $|S| < 2 \cdot \mathrm{OPT}$ by Theorem 2 and Corollary 1, against the (unknown) true optimum.

6.2.2. Results

Table 5 reports, for every instance, the category, the best-known approximate cover size (where published) or "–", the cover size produced by Hvala, the wall-clock solve time, and the resulting approximation ratio ("–" when no reference is available). Instances are listed alphabetically.

Table 5. Hvala on 130 real-world large graphs from the Network Data Repository [13]. The "Best Known" column gives the previously published best-known approximate cover size where one is available (source: Milagro [16]); "–" indicates no public reference value. By Theorem 2 and Corollary 1, every reported cover size is strictly less than $2 \cdot \mathrm{OPT}$.

Instance	Category	Known OPT	Hvala size	Time	Ratio
bio-celegans	Bio	248	257	30.3ms	1.036
bio-diseasome	Bio	283	285	18.7ms	1.007
bio-dmela	Bio	–	2672	495.3ms	–
bio-yeast	Bio	453	464	57.5ms	1.024
ca-AstroPh	Collab	–	11512	6.05s	–
ca-citeseer	Collab	–	129274	22.44s	–
ca-coauthors-dblp	Collab	–	472272	757.0s	–
ca-CondMat	Collab	–	12500	4.02s	–
ca-CSphd	Collab	548	553	79.1ms	1.009
ca-dblp-2010	Collab	–	122072	28.83s	–
ca-dblp-2012	Collab	–	165085	31.50s	–
ca-Erdos992	Collab	459	461	142.1ms	1.004
ca-GrQc	Collab	–	2213	254.4ms	–
ca-HepPh	Collab	–	6568	49.94s	–
ca-MathSciNet	Collab	–	140428	41.45s	–
ca-netscience	Collab	212	214	40.1ms	1.009
ia-email-EU	Email	–	820	1.50s	–
ia-email-univ	Email	603	609	124.4ms	1.010
ia-enron-large	Social	–	12820	6.52s	–
ia-enron-only	Social	86	87	21.0ms	1.012
ia-fb-messages	Social	578	593	111.6ms	1.026
ia-infect-dublin	Social	295	295	47.3ms	1.000
ia-infect-hyper	Social	91	93	60.3ms	1.022

 doi:10.20944/preprints202506.0875.v13

Table 5 – continued

Instance	Category	Known OPT	Hvala size	Time	Ratio
ia-reality	Social	–	81	123.2ms	–
ia-wiki-Talk	Wiki	–	17407	16.52s	–
inf-power	Infra	–	2267	291.9ms	–
inf-roadNet-CA	Infra	–	1058991	122.5s	–
inf-roadNet-PA	Infra	–	587209	72.8s	–
rec-amazon	Rec	–	48622	5.36s	–
rt-retweet	Retweet	31	32	5.2ms	1.032
rt-retweet-crawl	Retweet	–	81211	143.8s	–
rt-twitter-copen	Retweet	235	238	42.9ms	1.013
sc-msdoor	SciComp	–	382184	400.1s	–
sc-nasasrb	SciComp	–	51559	65.1s	–
sc-pkustk11	SciComp	–	84149	111.2s	–
sc-pkustk13	SciComp	–	89759	124.6s	–
sc-pwtk	SciComp	–	208297	221.8s	–
sc-shipsec1	SciComp	–	119415	82.9s	–
sc-shipsec5	SciComp	–	148790	99.6s	–
scc_enron-only	SCC	137	138	197.9ms	1.007
scc_fb-forum	SCC	370	372	1.96s	1.005
scc_fb-messages	SCC	–	1072	27.78s	–
scc_infect-dublin	SCC	–	9124	8.70s	–
scc_infect-hyper	SCC	109	110	155.0ms	1.009
scc_reality	SCC	–	2486	193.9s	–
scc_retweet	SCC	–	564	1.02s	–
scc_retweet-crawl	SCC	–	8435	492.2ms	–
scc_rt_alwefaq	SCC	35	35	7.6ms	1.000
scc_rt_assad	SCC	16	16	3.3ms	1.000
scc_rt_bahrain	SCC	37	37	2.9ms	1.000
scc_rt_barackobama	SCC	29	29	3.3ms	1.000
scc_rt_damascus	SCC	15	15	1.1ms	1.000
scc_rt_dash	SCC	15	15	1.1ms	1.000
scc_rt_gmanews	SCC	46	46	15.2ms	1.000
scc_rt_gop	SCC	6	6	0.0ms	1.000
scc_rt_http	SCC	2	2	0.0ms	1.000
scc_rt_israel	SCC	11	11	0.0ms	1.000
scc_rt_justinbieber	SCC	26	26	5.2ms	1.000
scc_rt_ksa	SCC	12	12	0.5ms	1.000
scc_rt_lebanon	SCC	5	5	0.0ms	1.000
scc_rt_libya	SCC	12	12	1.3ms	1.000
scc_rt_lolgop	SCC	103	103	52.3ms	1.000
scc_rt_mittromney	SCC	42	42	1.6ms	1.000
scc_rt_obama	SCC	4	4	0.0ms	1.000
scc_rt_occupy	SCC	22	22	1.1ms	1.000
scc_rt_occupywallstnyc	SCC	45	45	12.1ms	1.000
scc_rt_oman	SCC	6	6	0.0ms	1.000
scc_rt_onedirection	SCC	29	29	4.0ms	1.000
scc_rt_p2	SCC	12	12	0.0ms	1.000
scc_rt_qatif	SCC	5	5	0.0ms	1.000
scc_rt_saudi	SCC	17	17	1.0ms	1.000

 doi:10.20944/preprints202506.0875.v13

Table 5 – continued

Instance	Category	Known OPT	Hvala size	Time	Ratio
scc_rt_tcot	SCC	12	12	1.0ms	1.000
scc_rt_tlot	SCC	6	6	0.6ms	1.000
scc_rt_uae	SCC	8	8	1.0ms	1.000
scc_rt_voteonedirection	SCC	4	4	0.0ms	1.000
scc_twitter-copen	SCC	–	1328	20.24s	–
soc-BlogCatalog	Social	–	20967	69.1s	–
soc-brightkite	Social	–	21473	10.30s	–
soc-buzznet	Social	–	31059	93.6s	–
soc-delicious	Social	–	86810	48.30s	–
soc-digg	Social	–	104237	217.9s	–
soc-dolphins	Social	34	35	3.2ms	1.029
soc-douban	Social	–	8685	24.07s	–
soc-epinions	Social	–	9858	3.09s	–
soc-flickr	Social	–	154387	107.8s	–
soc-flixster	Social	–	96404	283.6s	–
soc-FourSquare	Social	–	90524	127.9s	–
soc-gowalla	Social	–	85360	35.31s	–
soc-karate	Social	14	14	1.1ms	1.000
soc-lastfm	Social	–	78832	164.7s	–
soc-LiveMocha	Social	–	44146	79.9s	–
soc-slashdot	Social	–	22632	16.07s	–
soc-twitter-follows	Social	–	2323	24.34s	–
soc-wiki-Vote	Social	404	410	39.8ms	1.015
soc-youtube	Social	–	148135	64.9s	–
soc-youtube-snap	Social	–	279062	100.8s	–
socfb-Berkeley13	Facebook	–	17487	35.10s	–
socfb-CMU	Facebook	–	5061	8.45s	–
socfb-Duke14	Facebook	–	7790	15.06s	–
socfb-Indiana	Facebook	–	23741	44.05s	–
socfb-MIT	Facebook	–	4726	8.26s	–
socfb-OR	Facebook	–	37209	25.68s	–
socfb-Penn94	Facebook	–	31723	48.15s	–
socfb-Stanford3	Facebook	–	8611	19.07s	–
socfb-Texas84	Facebook	–	28669	55.17s	–
socfb-UCLA	Facebook	–	15494	24.95s	–
socfb-UConn	Facebook	–	13436	18.95s	–
socfb-UCSB37	Facebook	–	11481	14.06s	–
socfb-UF	Facebook	–	27775	52.03s	–
socfb-UIllinois	Facebook	–	24465	40.99s	–
socfb-Wisconsin87	Facebook	–	18716	28.95s	–
tech-as-caida2007	Tech	–	3699	1.07s	–
tech-as-skitter	Tech	–	529662	365.1s	–
tech-internet-as	Tech	–	5718	1.81s	–
tech-p2p-gnutella	Tech	–	15730	3.53s	–
tech-RL-caida	Tech	–	75568	14.69s	–
tech-routers-rf	Tech	793	801	94.7ms	1.010
tech-WHOIS	Tech	–	2297	964.5ms	–
web-arabic-2005	Web	–	115297	62.7s	–

 doi:10.20944/preprints202506.0875.v13

Table 5 – continued

Instance	Category	Known OPT	Hvala size	Time	Ratio
web-BerkStan	Web	–	5404	336.0ms	–
web-edu	Web	1449	1451	90.4ms	1.001
web-google	Web	497	498	40.3ms	1.002
web-indochina-2004	Web	–	7363	778.7ms	–
web-it-2004	Web	–	415230	182.0s	–
web-polblogs	Web	243	245	28.2ms	1.008
web-sk-2005	Web	–	58411	6.32s	–
web-spam	Web	–	2344	574.6ms	–
web-uk-2005	Web	–	127774	316.9s	–
web-webbase-2001	Web	–	2665	425.0ms	–
web-wikipedia2009	Web	–	659409	192.1s	–

6.2.3. Summary Statistics

Across the 130 real-world instances, Hvala achieves:

- **Approximation ratio (on the 51 instances with known best-known optima):**
  - Mean approximation ratio: **1.006**.
  - Minimum ratio: 1.000 (reached on 30 instances).
  - Maximum ratio: 1.036, on `bio-celegans` (*C. elegans* metabolic network; Hvala size 257 vs. best-known 248).
  - Distribution: all 51 ratios lie below 1.05; in particular, below 1.10 and far below the $\sqrt{2} \approx 1.414$ hardness threshold.
- **Scale:** the largest instance by vertex count is `soc-flixster`, a movie-rating graph with 2,523,386 vertices and 7,918,801 edges, solved in 4.73 minutes; the largest by edge count is `ca-coauthors-dblp` with 540,486 vertices and 15,245,729 edges, solved in 12.62 minutes (the longest solve of the experiment). Other large instances include `tech-as-skitter` (1,694,616 vertices, 11,094,209 edges, 6.08 min), `web-wikipedia2009` (1,864,433 vertices, 4,507,315 edges, 3.20 min), and `inf-roadNet-CA` (1,957,027 vertices, 2,760,388 edges, 2.04 min).
- **Runtime distribution:** 60 of the 130 instances are solved in under 1 second; 43 between 1 and 60 seconds; 26 between 1 and 10 minutes; exactly one (`ca-coauthors-dblp`) between 10 and 60 minutes; none exceed one hour.
- **Total wall-clock time:** cumulative solve time across all 130 real-world instances is approximately 5,732 seconds ($\approx$ 95.5 minutes, or 1.59 hours).
- **Linear-time scalability:** per-vertex amortised cost stays within a narrow range across five orders of magnitude of graph size, consistent with the $\mathcal{O}(n+m)$ complexity established in Theorem 1.

A linear-time algorithm that solves all 130 instances of a standard real-world benchmark — including multi-million-vertex graphs — in under 100 minutes total, with mean ratio 1.006, worst ratio 1.036, and every returned cover provably within a factor strictly less than 2 of the optimum, is the central practical takeaway of this section.

7. Discussion

7.1. Empirical vs. Theoretical Gap

Theorem 2 and Corollary 1 give a uniform ≤ 2 bound and a pointwise strict < 2 bound respectively, with the supremum of the ratio over all graphs equal to 2. Across the two experimental studies (Section 6.1 and Section 6.2, 239 instances in total), the empirical ratios on the 160 instances with known optima are far below this worst-case: combined mean 1.016, combined maximum 1.192 (on a single adversarially-constructed Sanchis instance), and every ratio strictly below $\sqrt{2} \approx 1.414$. The

 doi:10.20944/preprints202506.0875.v13

gap between the proved worst-case behaviour (ratios asymptotically reaching 2) and the observed ratios on real benchmarks is large, and closing it — either by refining the analysis to bound the ratio as a function of graph parameters (average degree, girth, treewidth, clique number) or by constructing adversarial instances that drive Hvala close to 2 — is a natural target for further work. We note in particular that the corrected runtime for `inf-roadNet-CA` (1,957,027 vertices, 122.50 seconds = 2.04 minutes) is consistent with the $\mathcal{O}(n+m)$ scaling observed across the full benchmark.

7.2. Hardness Barriers

The hardness results surveyed in the introduction [5–8,10] make it clear that no *unconditional* polynomial-time algorithm is known to achieve uniform constant ratio below $2-\epsilon$ for any fixed $\epsilon > 0$, and ratio below $\sqrt{2}$ is SETH-hard. Hvala does not aim to cross these barriers; it aims to match the ≤ 2 bound constructively, in linear time, and to inherit the pointwise strict < 2 inequality from the Hallelujah heuristic of the companion paper [11]. The ensemble and pruning are engineered to exploit structural orthogonality empirically, which accounts for the mean approximation ratio of 1.021 on the 109 NPBench instances with known optima and the combined mean ratio of 1.016 across all 160 known-optimum instances (NPBench and real-world combined), without contradicting any hardness result.

7.3. Prospects for a $\sqrt{2}-\epsilon$ Bound

The most interesting empirical regularity across both experimental studies is that *every single ratio observed on the* 160 *instances with known optima stays below* 1.414 — with the combined maximum being 1.192, on a narrow family of Sanchis adversarial graphs; 93.8% of the 160 instances are within ratio 1.05, 96.9% are within ratio 1.10, and 100% are within ratio 1.20. We stress the numerical threshold 1.414 rather than $\sqrt{2}$ deliberately: the question we wish to pose is whether the ratio of Hvala can be bounded uniformly by $\sqrt{2}-\epsilon$ for a *fixed* constant $\epsilon > 0$, not whether it is merely strictly below $\sqrt{2}$ in the same asymptotic-to-a-threshold sense that our inherited bound is asymptotic to 2.

Under SETH and the hardness results of Khot, Minzer and Safra [6–8], no polynomial-time algorithm can achieve uniform ratio $\sqrt{2}-\epsilon$ for any fixed $\epsilon > 0$ on *all* finite graphs unless P = NP. Hvala's empirical behaviour therefore cannot, on its own, imply a uniform $\sqrt{2}-\epsilon$ guarantee on all graphs. What it does suggest, in our view, is that Hvala (and more specifically the Hallelujah weighted-reduction component at its core [11]) is a plausible candidate for a refined worst-case analysis aimed at establishing a uniform $\sqrt{2}-\epsilon$ bound with fixed $\epsilon > 0$ on *restricted but broad* graph classes — for instance, graphs of bounded maximum degree, graphs with bounded clique number, graphs with bounded treewidth, or graphs drawn from structural families (power-law, expander-like, or small-world) that are common in practice. Such a restricted-class result would not contradict any known hardness barrier, and would be of substantial theoretical and practical interest.

Three observations support this interpretation. First, the 18 instances solved to exact optimality on NPBench are concentrated in highly-structured families (`c-fat`, `hamming`, `johnson`, `MANN_a9`, `p_hat300-1`), indicating that the Hallelujah reduction captures optimal structure on graphs where degree signals are uninformative but regularity is high. Second, the worst-case ratios on NPBench occur exclusively on the Sanchis `san/gen` hidden-clique adversarial construction, a narrow and specifically engineered graph family; on no other NPBench family does Hvala exceed ratio 1.08. Third, on the 130 real-world large graphs — including social, collaboration, web, biological, infrastructure, and scientific-computing networks at scales up to 2.5 million vertices and 15 million edges — Hvala's output always sits strictly below $2 \cdot \mathrm{OPT}$, the algorithm's linear-time scaling holds in practice, demonstrating that the ensemble can tighten, not just approximate, standing reference values.

We therefore position this paper as a step towards, rather than a proof of, a uniform $\sqrt{2}-\epsilon$ guarantee (with fixed $\epsilon > 0$) on restricted graph classes. We do *not* claim a uniform $\sqrt{2}-\epsilon$ bound here: such a claim would need to be accompanied by a proof, and no such proof is provided. What we claim is that Hvala is the first simple linear-time algorithm for Minimum Vertex Cover whose combined theoretical properties (rigorous ≤ 2, pointwise strict < 2, asymptotic-to-2 supremum) and empirical

 doi:10.20944/preprints202506.0875.v13

behaviour (ratios staying below 1.414 across 239 diverse instances, 160 of which with known optima, and cumulative real-world solve time of approximately 95.5 minutes for 130 instances reaching up to 2.5 million vertices) jointly make it a plausible vehicle for further theoretical work on the $\sqrt{2} - \epsilon$ threshold.

7.4. Comparison to Other Practical Methods

Advanced local-search methods such as FastVC [17], TIVC [18], and MetaVC2 [19] reach empirical ratios comparable to Hvala's on DIMACS-style benchmarks, typically at the price of longer run times and without a simple constructive worst-case guarantee. Parameterised FPT algorithms [20] are exact for small solution sizes k, complementing rather than competing with Hvala's regime of large, general graphs. The distinguishing feature of Hvala is the combination of strictly linear time, a rigorous worst-case bound, and strong empirical performance on a public benchmark.

8. Conclusions

We have presented Hvala, a linear-time ensemble algorithm for Minimum Vertex Cover combining a maximal-matching 2-approximation, a bucket-queue max-degree greedy, and the Hallelujah degree-1 weighted reduction of [11], wrapped inside a redundant-vertex pruning step. We proved rigorously that the algorithm achieves the uniform worst-case ratio $\rho \leq 2$ (Theorem 2) and, combining with the companion paper [11], the strict pointwise inequality $|S| < 2 \cdot \mathrm{OPT}(G)$ on every finite simple graph (Corollary 1) — the ratio is asymptotic to 2: strictly less than 2 on each graph, with supremum equal to 2 over all graphs. Hvala runs in $\mathcal{O}(n+m)$ time and $\mathcal{O}(n+m)$ space (Theorem 1).

We validated Hvala on *two* independent experimental studies totalling 239 instances. On the 109 instances with known optima from the NPBench vertex-cover collection (Experiment 1, Section 6.1, 126.97 seconds cumulative), Hvala solves 18 to proven optimality and attains a mean approximation ratio of 1.021. On the 130 real-world large graphs from the Network Data Repository (Experiment 2, Section 6.2), ranging up to 2,523,386 vertices and 15,245,729 edges, Hvala completes the entire benchmark in approximately 5,732 seconds ($\approx$ 95.5 minutes) cumulative — 60 of the 130 instances finish in under 1 second, and the largest instance (`ca-coauthors-dblp`, 540,486 vertices, 15,245,729 edges) is solved in 12.62 minutes — every returned cover is guaranteed by Corollary 1 to be strictly less than $2 \cdot \mathrm{OPT}$.

Across both studies, every single approximation ratio observed on the 160 instances with known optima stays below 1.414, with the maximum being 1.192 on a narrow family of Sanchis adversarial graphs. This empirical regularity — a hard ceiling at 1.414 across 239 structurally diverse instances — motivates the central open problem we propose as the natural continuation of this work:

> *Is there a fixed constant $\epsilon > 0$ such that, for every finite simple undirected graph G, the Hvala algorithm achieves approximation ratio $|S|/\mathrm{OPT}(G) \leq \sqrt{2} - \epsilon \approx 1.414 - \epsilon$ — or, failing that, does such a uniform bound hold on broad but restricted graph classes (bounded degree, bounded clique number, bounded treewidth, or structural families such as power-law and expander-like graphs)?*

We stress that the conjectured bound is of the form $\sqrt{2} - \epsilon$ for a *fixed* constant $\epsilon > 0$, not an asymptotic $< \sqrt{2}$: an asymptotic-to-$\sqrt{2}$ bound, in the same sense that our inherited bound is asymptotic-to-2, would not constitute a meaningful breakthrough. A fixed-constant $\sqrt{2} - \epsilon$ bound, in contrast, would either yield a uniform sub-$\sqrt{2}$ guarantee on all graphs (which, by the SETH-based hardness of Khot, Minzer and Safra [6–8], would imply P = NP) or, more realistically, a uniform fixed-constant guarantee on a specific restricted class — a result that would be of substantial theoretical and practical interest on its own.

We do *not* prove such a fixed-constant $\sqrt{2} - \epsilon$ bound in this paper, and we do not claim one holds on all graphs. What we claim is that Hvala is the first simple linear-time algorithm for Minimum Vertex Cover whose combined theoretical and empirical profile — rigorous ≤ 2 bound, pointwise strict < 2, linear time, and observed ratios uniformly below 1.414 across 239 diverse instances — makes the question above a plausible and well-posed target for future work.

 doi:10.20944/preprints202506.0875.v13

Availability. The Hvala algorithm is distributed via PyPI:

- **Package:** https://pypi.org/project/hvala
- **Installation:** `pip install hvala`
- **Usage:** `from hvala.algorithm import find_vertex_cover`

Acknowledgment

The author is sincerely grateful to Iris, Marilin, Sonia, Yoselin, Arelis, Anissa, Liuva, Yudit, Gretel, Gema, and Blaquier, as well as Israel, Arderi, Juan Carlos, Yamil, Alejandro, Aroldo, Yary, Reinaldo, Alex, Emmanuel, and Michael for their constant support. Whether through encouragement, stimulating conversations, practical assistance, or simply being present during challenging moments, their contributions have played an important role in bringing this work to completion.

References

1. Karp, R.M. Reducibility Among Combinatorial Problems. In *50 Years of Integer Programming 1958–2008: From the Early Years to the State-of-the-Art*; Springer: Berlin, Germany, 2010; pp. 219–241. https://doi.org/10.1007/978-3-540-68279-0_8.
2. Papadimitriou, C.H.; Steiglitz, K. *Combinatorial Optimization: Algorithms and Complexity*; Courier Corporation: North Chelmsford (MA), 1998.
3. Karakostas, G. A Better Approximation Ratio for the Vertex Cover Problem. *ACM Transactions on Algorithms* **2009**, *5*, 1–8. https://doi.org/10.1145/1597036.1597045.
4. Karpinski, M.; Zelikovsky, A. Approximating Dense Cases of Covering Problems. In Proceedings of the DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Providence, Rhode Island, 1996; Vol. 26, pp. 147–164.
5. Dinur, I.; Safra, S. On the Hardness of Approximating Minimum Vertex Cover. *Annals of Mathematics* **2005**, *162*, 439–485. https://doi.org/10.4007/annals.2005.162.439.
6. Khot, S.; Minzer, D.; Safra, M. On Independent Sets, 2-to-2 Games, and Grassmann Graphs. In Proceedings of the Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, Montreal, Québec, Canada, 2017; pp. 576–589. https://doi.org/10.1145/3055399.3055432.
7. Dinur, I.; Khot, S.; Kindler, G.; Minzer, D.; Safra, M. Towards a proof of the 2-to-1 games conjecture? In Proceedings of the Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, Los Angeles, California, 2018; pp. 376–389. https://doi.org/10.1145/3188745.3188804.
8. Khot, S.; Minzer, D.; Safra, M. Pseudorandom Sets in Grassmann Graph Have Near-Perfect Expansion. In Proceedings of the 2018 IEEE 59th Annual Symposium on Foundations of Computer Science, Paris, France, 2018; pp. 592–601. https://doi.org/10.1109/FOCS.2018.00062.
9. Khot, S. On the Power of Unique 2-Prover 1-Round Games. In Proceedings of the Proceedings of the 34th Annual ACM Symposium on Theory of Computing, Montreal, Québec, Canada, 2002; pp. 767–775. https://doi.org/10.1145/509907.510017.
10. Khot, S.; Regev, O. Vertex Cover Might Be Hard to Approximate to Within $2-\epsilon$. *Journal of Computer and System Sciences* **2008**, *74*, 335–349. https://doi.org/10.1016/j.jcss.2007.06.019.
11. Vega, F. An Approximate Solution to the Minimum Vertex Cover Problem: The Hallelujah Algorithm. *International Journal of Parallel, Emergent and Distributed Systems* **2026**. Accepted for publication. Permanent link: https://doi.org/10.1080/17445760.2026.2660724, https://doi.org/10.1080/17445760.2026.2660724.
12. Nguyen, T.; Bui, T. NP-Complete Benchmark Instances. https://roars.dev/npbench/. Vertex cover benchmark collection; FRB instances (Ke Xu), DIMACS clique complements, random graphs (Periannan).
13. Rossi, R.; Ahmed, N. The Network Data Repository with Interactive Graph Analytics and Visualization, Palo Alto (CA), 2015; Vol. 29. https://doi.org/10.1609/aaai.v29i1.9277.
14. Vega, F. Hvala: Approximate Vertex Cover Solver. https://pypi.org/project/hvala, 2026. Accessed: 2026-04-22.
15. Mascia, F. The Maximum Clique Problem – DIMACS Benchmark Set. https://iridia.ulb.ac.be/~fmascia/maximum_clique/DIMACS-benchmark, 2015. Compiled clique-number values for DIMACS Second Implementation Challenge instances.
16. Vega, F. The Milagro Experiment. https://github.com/frankvegadelgado/milagro, 2026. Accessed: 2026-04-20.

 doi:10.20944/preprints202506.0875.v13

17. Cai, S.; Lin, J.; Luo, C. Finding a Small Vertex Cover in Massive Sparse Graphs. *Journal of Artificial Intelligence Research* **2017**, *59*, 463–494. https://doi.org/10.1613/jair.5443.
18. Zhang, Y.; Wang, S.; Liu, C.; Zhu, E. TIVC: An Efficient Local Search Algorithm for Minimum Vertex Cover in Large Graphs. *Sensors* **2023**, *23*, 7831. https://doi.org/10.3390/s23187831.
19. Luo, C.; Hoos, H.H.; Cai, S.; Lin, Q.; Zhang, H.; Zhang, D. Local search with efficient automatic configuration for minimum vertex cover. In Proceedings of the Proceedings of the 28th International Joint Conference on Artificial Intelligence, Macao, China, 2019; pp. 1297–1304.
20. Harris, D.G.; Narayanaswamy, N.S. A Faster Algorithm for Vertex Cover Parameterized by Solution Size. In Proceedings of the 41st International Symposium on Theoretical Aspects of Computer Science, Clermont-Ferrand, France, 2024; Vol. 289, pp. 40:1–40:18. https://doi.org/10.4230/LIPIcs.STACS.2024.40.