The Esperanza Algorithm for Maximum Independent Set

Frank Vega

Information Physics Institute, 840 W 67th St, Hialeah, FL 33012, USA

hexagon:2610.00703v1

Abstract

The maximum independent set problem asks for the largest set of pairwise non-adjacent vertices in an undirected graph. It is NP-hard to approximate within $n^{1-\varepsilon}$ for every $\varepsilon > 0$, and under the Unique Games Conjecture it is hard to approximate within $O(\Delta/\log^2 \Delta)$ on graphs of maximum degree Δ; the Khot–Regev form of the conjecture, for which a proof has been reported but not yet peer reviewed, also rules out a $(2-\varepsilon)$-approximation of Minimum Vertex Cover. We present version 0.1.6 of Esperanza, which takes the better of a dynamic minimum-degree (Caro–Wei) greedy set and the complement of the Salvador vertex cover, and improves it by three budgeted local searches: repair-and-grow restarts at the cover vertices and at a maximum-degree vertex, chained local restarts realising $(k, k+1)$-improvements, and an iterated $(1,2)$-swap search. We prove that it runs in $O((n+m)\,\alpha_{\mathrm{A}}(n))$ expected time for fixed budgets, with α_{A} the inverse Ackermann function. We prove that the output is a maximal independent set, that its complementary vertex cover has at most $2\tau - r$ vertices, where τ is the minimum cover size and r the number of components with an edge, and that the ratio is at most $(\Delta+2)/3$ on graphs of maximum degree Δ. We prove the ratio $2\sqrt{n}$ on every graph with $\Delta \leq \sqrt{n}$ and on three further classes, and state it for all graphs as a conjecture, whose proof would imply P = NP. We also state the conjecture that the complementary cover has at most $2\tau - |M|$ vertices for the NetworkX minimum maximal matching M; a proof would give a 3/2-approximation for Minimum Vertex Cover and refute the Unique Games Conjecture unless P = NP. In reproducible experiments the returned set is optimal on all 30 000 graphs of a suite certified by an exact integer program; on 40 Khot–Regev YES instances with 640 to 2 560 vertices the conjectured bound is certified on 38 and open on 2; and on 31 bounded-degree graphs the certified ratio is at most 1.0769 on the families with a planted independent set.

Keywords: Maximum Independent Set, Approximation Algorithm, Vertex Cover, Local Search, Bounded-degree Graphs, Linear Time, Unique Games Conjecture, P versus NP
2020 MSC: 05C69, 68Q25, 68W25, 90C27

1. Introduction

Given an undirected graph $G = (V, E)$, an *independent set* is a subset $S \subseteq V$ in which no two vertices are joined by an edge. The maximum independent set (MIS) problem asks for such a set of largest possible cardinality; its size is the independence number $\alpha(G)$. A set $C \subseteq V$ is a *vertex cover* if every edge has an endpoint in C; C is a vertex cover if and only if $V \setminus C$ is independent, so the minimum cover size is $\tau(G) = n - \alpha(G)$. We write $n = |V|$, $m = |E|$, Δ for the maximum degree, and $r(G)$ for the number of connected components of G that contain an edge. The ratio of a returned independent set S is $\rho = \alpha(G)/|S|$. MIS is on Karp's list of NP-complete problems [1].

On general graphs, MIS cannot be approximated within $n^{1-\varepsilon}$ for any $\varepsilon > 0$ unless NP = ZPP [2], and the same threshold is NP-hard [3]. The best known polynomial-time ratios are $O(n/(\log n)^2)$ by subgraph exclusion [4] and $O(n(\log\log n)^2/(\log n)^3)$ [5]. On graphs of maximum degree Δ, the minimum-degree greedy algorithm has ratio $(\Delta+2)/3$ [6], and the best known ratio, due to Halperin [7], is $O(\Delta \log\log \Delta/\log \Delta)$. Under the Unique Games Conjecture (UGC) [8], Austrin, Khot and Safra proved that MIS on graphs of maximum degree Δ is hard to approximate within $O(\Delta/\log^2 \Delta)$ [9], and Chan proved that $O(\Delta/\log^4 \Delta)$ is NP-hard without any conjecture [10]. Khot and Regev

Email address: vega.frank@gmail.com (Frank Vega)

proved, under UGC, that for every $\varepsilon, \delta > 0$ it is hard to distinguish graphs with an independent set of $(\frac{1}{2} - \varepsilon)n$ vertices from graphs with no independent set of more than δn vertices; equivalently, Minimum Vertex Cover is hard to approximate within $2 - \varepsilon$ for every $\varepsilon > 0$ [11]. In October 2026 the OpenAI Math Team released a manuscript with Lean artifacts reporting a proof of UGC in the NP-hardness form that the theorem of Khot and Regev requires [12]; at the time of writing it has not been peer reviewed. If it holds, both hardness results above hold unless P = NP.

This paper describes version 0.1.6 of Esperanza, which is implemented in Python [13]. The algorithm takes the best of a dynamic minimum-degree (Caro–Wei) greedy set [14, 15, 6] and the complement of the vertex cover returned by Salvador [16, 17], and improves it by three budgeted local searches: a repair-and-grow restart at every cover vertex and at a maximum-degree vertex, chained local restarts that realise $(k, k+1)$-improvements, and an iterated $(1, 2)$-swap local search [18].

Contributions. The contributions of this paper are:

- The algorithm (Section 3, Algorithms 1–4) and the proof that it returns a maximal independent set (Theorem 4.5).
- A near-linear running time: for fixed work budgets the algorithm runs in $O((n + m)\,\alpha_{\mathrm{A}}(n))$ expected time, with α_{A} the inverse Ackermann function (Theorem 7.2).
- Proved bounds (Section 5): the complementary vertex cover $V \setminus S$ is never larger than the Salvador cover, so $|V \setminus S| \le 2\tau(G) - r(G)$ on every graph without self-loops (Theorem 5.1); $|S| \ge 2\alpha(G) - n + r(G)$ (Corollary 5.2); and $|S| \ge \sum_v 1/(\deg v + 1)$ with ratio at most $(\Delta + 2)/3$ on graphs of maximum degree Δ (Theorem 5.3); and the ratio $2\sqrt{n}$ on every graph with $\Delta \le \sqrt{n}$ and on three further classes (Theorem 5.4).
- The conjecture that the ratio is at most $2\sqrt{n}$ on every graph (Conjecture 6.6), with the theorem that a proof would imply P = NP (Theorem 6.7).
- The conjecture that $|V \setminus S| \le 2\tau(G) - |M|$ for the NetworkX minimum maximal matching M (Conjecture 6.1), which is implied by the corresponding conjecture for Salvador, a per-instance certificate for it, and the conditional theorem that a proof would refute UGC unless P = NP, and would imply P = NP if the theorem reported in [12] holds (Theorem 6.5).
- Reproducible experiments (Section 8): 30 000 graphs with an optimum certified by an exact integer program, 40 YES-type instances of the Khot–Regev reduction with 640 to 2 560 vertices, and 31 bounded-degree graphs with Δ up to 128.

Section 3 describes the algorithm, Section 4 proves correctness, Section 5 the bounds, Section 6 states the conjecture and its consequences, and Section 7 the running time. Section 8 reports the experiments and Section 9 concludes.

2. Research Data

The algorithm is implemented in the Python package `esperanza`, available on the Python Package Index [13]. The version described in this paper is v0.1.6. Table 1 lists the code metadata.

Nr.	Code metadata description	Metadata
C1	Current code version	v0.1.6
C2	Permanent link to code/repository	`https://github.com/frankvegadelgado/esperanza`
C3	Permanent link to Reproducible Capsule	`https://pypi.org/project/esperanza/`
C4	Legal Code License	MIT License
C5	Code versioning system used	git
C6	Languages, tools, and services used	Python
C7	Compilation requirements and dependencies	Python ≥ 3.12, Salvador ≥ 0.1.4
C8	Experiment scripts (Section 8)	`car/` directory of the repository

Table 1: Code metadata for the `esperanza` package.

3. The Esperanza Algorithm

Algorithms 1–4 give the pseudo-code of Esperanza; they follow the published implementation (version 0.1.6 [13]). Throughout, G denotes the working graph obtained in Algorithm 1 after removing self-loops and isolated vertices; its parameters n, m and Δ are those of the working graph. For a set S and a vertex v, the *tightness* of v is $t_S(v) = |N(v) \cap S|$.

Algorithm 1 FindIndependentSet(G; B_R, B_L, B_I, σ)

Require: Undirected graph $G = (V, E)$; work budgets B_R, B_L, B_I (default 100 each); seed σ
Ensure: A maximal independent set of G

1: **if** $|V| = 0$ **then return** $\emptyset$
2: **end if**
3: $G \leftarrow G$.copy(); remove all self-loops from G
4: $I_{iso} \leftarrow \{v \in V : \deg_G(v) = 0\}$; remove I_{iso} from G
5: **if** $|V(G)| = 0$ **then return** I_{iso}
6: **end if**
7: $I_{CW} \leftarrow$ PureCaroWeiBaseline(G) ▷ Algorithm 2
8: $C \leftarrow$ Salvador(G) ▷ [16, 17]; Lemma 4.3
9: $S \leftarrow I_{CW}$ **if** $|I_{CW}| > |V(G) \setminus C|$ **else** $V(G) \setminus C$
10: $v_{max} \leftarrow$ a vertex of maximum degree in G
11: $I_R \leftarrow V(G) \setminus$ RefineCover(G, C, v_{max}, B_R) ▷ restarts from $C \cup \{v_{max}\}$
12: **if** $|I_R| > |S|$ **then** $S \leftarrow I_R$
13: **end if**
14: $I_L \leftarrow$ LocalRestartSearch(G, S, B_L) ▷ Algorithm 4
15: **if** $|I_L| > |S|$ **then** $S \leftarrow I_L$
16: **end if**
17: $I_I \leftarrow V(G) \setminus$ ImproveVertexCover($G, V(G) \setminus S, B_I, \sigma$) ▷ (1, 2)-swaps [18]
18: **if** $|I_I| > |S|$ **then** $S \leftarrow I_I$
19: **end if**
20: $S \leftarrow$ MaximizeSolution(G, S) ▷ Algorithm 3
21: **return** $S \cup I_{iso}$

The calls to Salvador. Three routines of the package `salvador` (version $\geq$ 0.1.4) [17] are used. Salvador(G) in line 8 is `salvador.algorithm.find_vertex_cover`, the final cover of the seventeen-strategy ensemble of [16]. RefineCover(G, C, v_{max}, B) in line 11 is `salvador.algorithm.refine_cover`: for every $u \in C \cup \{v_{max}\}$ it forms the vertex cover $(C \setminus \{u\}) \cup N(u)$, that is, it moves u into $V \setminus C$ and evicts the neighbours of u, grows the resulting independent set into a maximal one by MaximizeSolution, and keeps the smallest resulting cover. When C is a minimal cover, the routine evaluates each restart locally on the neighbours of the evicted vertices, which gives the same set as the global MaximizeSolution pass; every adjacency scan and the sorting of every local list are charged to a work counter, and the routine stops when the next restart would exceed $B(n + m)$. ImproveVertexCover(G, C', B, σ) in line 17 is `salvador.local_search.improve_vertex_cover`: it improves the independent set $V \setminus C'$ by the $(1, 2)$-swaps of Andrade, Resende and Werneck [18], which remove one solution vertex and insert two non-adjacent 1-tight neighbours of it, followed by an iterated local search that forces a random vertex into the set, re-inserts free vertices, runs $(1, 2)$-swaps around the change and undoes the step when the set shrinks. Its random choices use the fixed seed σ (`ILS_SEED` = 20261031 in the package), and its work is charged to a counter with limit $B(n + m)$.

Local restarts. Algorithm 4 applies the restart of RefineCover to the incumbent S instead of the Salvador cover. A move at $u \notin S$ forces u into S, evicts the $k = t(u)$ vertices of $X = N(u) \cap S$, and refills the freed region with every vertex $w \in N(X)$ that has no neighbour left in S, in ascending order of degree. The move is committed when it inserts at least k vertices besides u, a $(k, k + 1)$-improvement, and undone otherwise; committed moves requeue the vertices whose tightness decreased, so improvements are chained.

Algorithm 2 PureCaroWeiBaseline(G): dynamic minimum-degree greedy

Require: Undirected graph $G = (V, E)$
Ensure: A maximal independent set I with $|I| \geq \sum_v 1/(\deg_G(v) + 1)$

```
deg[v] ← deg_G(v) for every v; place every v into bucket B[deg[v]]; ptr ← 0
removed ← ∅; I ← ∅
while some vertex is not removed do
    advance ptr to the first bucket whose front is not removed, discarding removed fronts
    v ← pop front of B[ptr]; I ← I ∪ {v}; mark v removed
    for each neighbour u of v that is not removed do
        mark u removed
        for each neighbour w ≠ v of u that is not removed do
            deg[w] ← deg[w] − 1; append w to B[deg[w]]
            if deg[w] < ptr then ptr ← deg[w]
            end if
        end for
    end for
end while
return I
```

Algorithm 3 MaximizeSolution(G, S): repair and grow in $O(n + m)$

Require: Undirected graph $G = (V, E)$; candidate set $S \subseteq V$ (may contain conflicts)
Ensure: A maximal independent set of G

```
I ← S; conflicts[u] ← |N(u) ∩ I| for u ∈ I
for each edge {u, v} ∈ E do                          ▷ Phase 1: conflict resolution
    if u ∈ I and v ∈ I then
        ℓ ← u if conflicts[u] ≥ conflicts[v] else v
        I ← I \ {ℓ}; decrement conflicts[w] for every w ∈ N(ℓ) ∩ I
    end if
end for
place every u ∉ I into the bucket B[deg_G(u)]        ▷ Phase 2: growth by counting sort
for d = 0, ..., Δ do
    for each u ∈ B[d] do
        if N(u) ∩ I = ∅ then I ← I ∪ {u}
        end if
    end for
end for
return I
```

Algorithm 4 LocalRestartSearch(G, S, B): budgeted local restarts

Require: Graph G without self-loops; independent set S; budget B
Ensure: An independent set S' with $|S'| \geq |S|$

maintain $t(v) = |N(v) \cap S|$ for every v; work $\leftarrow$ cost of the initialisation; limit $\leftarrow$ work + $B(n+m)$ + 1
$Q \leftarrow$ the vertices outside S, sorted by $(t(v), \deg v)$ with two counting sorts
while $Q \neq \emptyset$ **and** work $\leq$ limit **do**
    $u \leftarrow Q$.PopFront(); work $\leftarrow$ work + 1
    if $u \in S$ **then continue**
    end if
    if $t(u) = 0$ **then** insert u into S; **continue**
    end if
    $X \leftarrow N(u) \cap S$; $T \leftarrow \sum_{x \in X} \deg x$; work $\leftarrow$ work + $\deg u$
    if work + $3(T + \deg u + |X| + 1) + T \log_2(T+1) >$ limit **then continue**
    end if
    remove every $x \in X$ from S; insert u into S ▷ logged
    $R \leftarrow \{w \in N(X) : w \notin S,\ t(w) = 0\}$, sorted by $(\deg w, w)$
    for each $w \in R$ **do**
        if $w \notin S$ **and** $t(w) = 0$ **then** insert w into S ▷ logged
        end if
    end for
    if S grew **then** append to Q every $w \in N(X) \setminus S$ not in Q ▷ commit
    else undo the logged operations
    end if
end while
return S

4. Correctness of the Algorithm

Lemma 4.1. *For every $S' \subseteq V(G)$,* MaximizeSolution(G, S') *returns a maximal independent set of G that contains every vertex of S' having no neighbour in S'. In particular, if S' is independent, then S' is contained in the output.*

Proof. During Phase 1 the set only shrinks. If an edge had both endpoints in the set at the end of Phase 1, both endpoints would have been present when that edge was scanned, and one of them would have been discarded; so the set is independent after Phase 1. A vertex of S' without neighbours in S' never lies on a scanned conflicting edge, so it is never discarded. Phase 2 adds a vertex only if it has no neighbour in the current set, so independence is preserved. Every vertex outside the set after Phase 1 is placed in a bucket and examined; if it is rejected, it has a neighbour in the set at that moment, and that neighbour is never removed. Hence the output is maximal. □

Lemma 4.2. *Algorithm 2 returns a maximal independent set I of G that is produced by the minimum-degree greedy rule: every chosen vertex has minimum degree in the graph of the vertices not yet removed. Consequently $|I| \geq \sum_{v \in V} \frac{1}{\deg_G(v)+1} \geq \frac{n}{\Delta+1}$.*

Proof. A chosen vertex is never adjacent to a previously chosen one, because all neighbours of a chosen vertex are removed at once; every vertex is either chosen or removed as a neighbour of a chosen vertex, so I is maximal. Degrees only decrease, and every vertex is appended to the bucket of its current degree whenever that degree changes, with the pointer moved back when a degree drops below it. Hence a vertex popped from bucket ptr that is not removed has current degree at most ptr, and every bucket below ptr contains only removed vertices; so the chosen vertex has minimum current degree. Let v be chosen when its current degree is d, and let $N[v]$ be its closed neighbourhood in the current graph. Every $u \in N[v]$ has current degree at least d and original degree at least its current degree, so $\sum_{u \in N[v]} \frac{1}{\deg_G(u)+1} \leq (d+1)\frac{1}{d+1} = 1$. The closed neighbourhoods of the chosen vertices partition V, so $\sum_{v \in V} \frac{1}{\deg_G(v)+1} \leq |I|$, and $\deg_G(v) \leq \Delta$ gives the second inequality. □

Lemma 4.3 (Properties of the Salvador routines)**.** *Let G be a graph without self-loops and isolated vertices.*

(i) The cover C of line 8 is a vertex cover with $|C| \le 2\tau(G) - r(G)$, computed in $O((n+m)\,\alpha_A(n))$ time [16].

(ii) RefineCover *and* ImproveVertexCover *return vertex covers no larger than their input cover, in $O(B(n+m)+n+m)$ time for a budget B [16, 17].*

Proof. (i) is the bound $2\tau - r$ and the running time proved for Salvador 0.1.4 in [16]. For (ii), RefineCover returns either its input or a cover of the form $V \setminus I$ with I independent and strictly larger than $V \setminus C$; ImproveVertexCover returns the complement of an independent set that is at least as large as the complement of its input, because worse steps are undone. The running times are those of the strategies c_{17} and c_{14} of [16], whose work counters stop after $B(n+m)$ units plus the cost of one step, which is $O(n+m)$. □

Lemma 4.4. *For an independent set S of a graph without self-loops, Algorithm 4 returns an independent set S' with $|S'| \ge |S|$.*

Proof. The tightness counters are updated on every insertion and removal, so $t(v) = |N(v) \cap S|$ holds at all times. A vertex is inserted only when its tightness is 0: a free vertex directly, the vertex u after all of $N(u) \cap S$ has been removed, and a refill vertex w after the check $t(w) = 0$. Hence S stays independent. A free insertion increases $|S|$. A move is kept only if 1 + (number of refill insertions) $> |X|$, that is, if $|S|$ increases; otherwise the log is undone in reverse order, which restores S and all counters. So $|S|$ never decreases. □

Theorem 4.5. *Algorithm 1 (*FindIndependentSet*) returns a maximal independent set of the input graph, where self-loops are ignored.*

Proof. I_{CW} is independent by Lemma 4.2. $V(G) \setminus C$, I_R and I_I are complements of vertex covers by Lemma 4.3, hence independent, and I_L is independent by Lemma 4.4. So the incumbent S is independent before the last call, and MaximizeSolution(G, S) returns a maximal independent set of the working graph by Lemma 4.1. Isolated vertices have no neighbours, so $S \cup I_{iso}$ is independent in the input graph; it is maximal, because a vertex outside it is a non-isolated vertex of the working graph and has a neighbour in S. □

5. Proved Bounds

Theorem 5.1 (The complementary cover)**.** *Let G be a graph without self-loops, S the set returned by Algorithm 1 and C the* Salvador *cover of line 8. Then $V \setminus S$ is a vertex cover of G and*

$$|V \setminus S| \;\le\; |C| \;\le\; 2\,\tau(G) - r(G).$$

Proof. $V \setminus S$ is a vertex cover because S is independent (Theorem 4.5). The incumbent starts with at least $|V(G) \setminus C|$ vertices and is only replaced by larger sets; the final MaximizeSolution call keeps the incumbent, which is independent, by Lemma 4.1. Hence $|S| \ge |V(G) \setminus C| + |I_{iso}|$, and $|V \setminus S| \le |C|$. Isolated vertices change neither τ nor r, so Lemma 4.3(i) gives $|C| \le 2\tau(G) - r(G)$. □

The package exposes this cover as `esperanza.algorithm.find_vertex_cover`, which also adds every vertex with a self-loop.

Corollary 5.2. *For every graph G without self-loops, $|S| \ge 2\alpha(G) - n + r(G)$. Hence, if $\alpha(G) = \beta n$ with $\beta > \frac{1}{2}$, then $\alpha(G)/|S| \le \beta/(2\beta - 1)$.*

Proof. By Theorem 5.1, $|S| \ge n - 2\tau(G) + r(G) = n - 2(n - \alpha(G)) + r(G)$. For $\alpha = \beta n$ with $\beta > \frac{1}{2}$ this gives $|S| \ge (2\beta - 1)n$. □

Theorem 5.3 (Bounded degree)**.** *For every graph G of maximum degree Δ, the set S returned by Algorithm 1 satisfies $|S| \ge \sum_v 1/(\deg v + 1) \ge n/(\Delta + 1)$ and*

$$\frac{\alpha(G)}{|S|} \;\le\; \frac{\Delta + 2}{3}.$$

Proof. The incumbent starts with at least $|I_{\mathrm{CW}}|$ vertices on the working graph and never shrinks (proof of Theorem 5.1), and isolated vertices are added to both S and every maximum independent set. I_{CW} is produced by the minimum-degree greedy rule (Lemma 4.2), which has ratio at most $(\Delta + 2)/3$ on graphs of maximum degree Δ [6], and satisfies the Caro–Wei bound by Lemma 4.2. □

Theorem 5.4 (Ratio $O(\sqrt{n})$ on four classes)**.** *Let G be a graph on n vertices without self-loops and isolated vertices, $\alpha = \alpha(G)$, C the* Salvador *cover of line 8 and S the set returned by Algorithm 1. If at least one of the following conditions holds, then $\alpha/|S| \le 2\sqrt{n}$:*

(a) $\alpha \le 2\sqrt{n}$;

(b) $2\alpha - n \ge \alpha/\sqrt{n}$;

(c) $|I^* \setminus C| \ge \alpha/\sqrt{n}$ *for some maximum independent set I^* of G;*

(d) $\Delta \le \sqrt{n}$.

Proof. By the proofs of Theorems 5.1 and 5.3, $|S| \ge |V \setminus C|$ and $|S| \ge |I_{\mathrm{CW}}|$. (a) S is non-empty, so $\alpha/|S| \le \alpha \le 2\sqrt{n}$. (b) $|C| \le 2\tau = 2(n - \alpha)$ by Lemma 4.3(i), so $|S| \ge n - |C| \ge 2\alpha - n \ge \alpha/\sqrt{n}$ and $\alpha/|S| \le \sqrt{n}$. (c) $I^* \setminus C \subseteq V \setminus C$, so $|S| \ge |I^* \setminus C| \ge \alpha/\sqrt{n}$ and $\alpha/|S| \le \sqrt{n}$. (d) By Lemma 4.2 and $\alpha \le n$,

$$|S| \;\ge\; |I_{\mathrm{CW}}| \;\ge\; \frac{n}{\Delta + 1} \;\ge\; \frac{n}{\sqrt{n} + 1} \;\ge\; \frac{\sqrt{n}}{2} \;\ge\; \frac{\alpha}{2\sqrt{n}},$$

where $\frac{n}{\sqrt{n}+1} \ge \frac{\sqrt{n}}{2}$ is equivalent to $n \ge \sqrt{n}$. Hence $\alpha/|S| \le 2\sqrt{n}$. □

The four conditions do not cover every graph. A graph satisfies none of them exactly when $\Delta > \sqrt{n}$, $2\sqrt{n} < \alpha$, $2\alpha - n < \alpha/\sqrt{n}$ and $|I^* \setminus C| < \alpha/\sqrt{n}$ for every maximum independent set I^*. In that case every maximal independent set has at least $n/(\Delta + 1)$ vertices, because each of its vertices dominates at most $\Delta + 1$ vertices, but $n/(\Delta + 1) < \sqrt{n}$, and this does not give $|S| \ge \alpha/(2\sqrt{n})$. The ratio $2\sqrt{n}$ on all graphs is therefore stated as Conjecture 6.6 in Section 6. On general graphs, by [2, 3], a polynomial-time ratio of $n^{1-\varepsilon}$ for some $\varepsilon > 0$ would imply P = NP.

The barrier $\Delta/\log^2 \Delta$. Austrin, Khot and Safra proved that, assuming the Unique Games Conjecture, Maximum Independent Set on graphs of maximum degree Δ is hard to approximate within a factor $O(\Delta/\log^2 \Delta)$ for all sufficiently large Δ [9]. Without any conjecture, Chan proved that the factor $O(\Delta/\log^4 \Delta)$ is NP-hard [10], and the best known polynomial-time ratio is $O(\Delta \log\log \Delta/\log \Delta)$ [7]. Consequently, a polynomial-time algorithm whose ratio on graphs of maximum degree Δ is $o(\Delta/\log^2 \Delta)$ as $\Delta \to \infty$ would refute the Unique Games Conjecture unless P = NP. The ratio $(\Delta + 2)/3$ proved in Theorem 5.3 is linear in Δ, so it lies above this threshold for large Δ and is consistent with both hardness results. Section 8.3 reports the measured ratios of Esperanza together with the value $\Delta/\log_2^2 \Delta$ for each instance; since the threshold is asymptotic and its constant is not explicit, finite instances do not decide whether the ratio of Esperanza grows more slowly than $\Delta/\log^2 \Delta$.

6. The Conjectured Bounds and Their Consequences

Let M be the maximal matching returned by NetworkX's `min_maximal_matching` [19] on the working graph; it is the matching of the strategy c_2 of Salvador [16]. Every vertex cover contains an endpoint of each of the $|M|$ disjoint edges of M, and the $2|M|$ matched vertices form a vertex cover, so $|M| \le \tau(G) \le 2|M|$.

Conjecture 6.1 (Esperanza bound)**.** *For every graph G without self-loops and isolated vertices, the set S returned by Algorithm 1 satisfies*

$$|V \setminus S| \;\le\; 2\,\tau(G) - |M|, \qquad \textit{equivalently} \qquad |V \setminus S| - \tau(G) \;\le\; \tau(G) - |M|.$$

Proposition 6.2. *Conjecture 6.1 is implied by the Ensemble-bound conjecture of [16], which states $|C| \le 2\tau(G) - |M|$ for the* Salvador *cover C. If Conjecture 6.1 holds, then $|V \setminus S| \le \frac{3}{2}\tau(G)$ for every graph.*

Proof. $|V \setminus S| \le |C|$ by Theorem 5.1. The second statement follows from $|M| \ge \tau(G)/2$. □

Proposition 6.3 (Per-instance certificate)**.** *Let $L \le \tau(G) \le U$ be a lower and an upper bound. If $|V \setminus S| \le 2L - |M|$, then Conjecture 6.1 holds on G; if $|V \setminus S| > 2U - |M|$, it fails on G. If $|V \setminus S| \le \frac{3}{2}L$, then $|V \setminus S| \le \frac{3}{2}\tau(G)$.*

Proof. $2L - |M| \le 2\tau - |M| \le 2U - |M|$, and $\frac{3}{2}L \le \frac{3}{2}\tau$. □

Proposition 6.4 (NO-type instances)**.** *If G has no independent set of more than δn vertices with $\delta \le \frac{1}{4}$, then every vertex cover C' of G satisfies $|C'| \le 2\tau(G) - |M|$. In particular, Conjecture 6.1 holds on the NO-type instances of the reduction of Khot and Regev [11] with $\delta \le \frac{1}{4}$.*

Proof. $\tau(G) = n - \alpha(G) \ge (1 - \delta)n$ and $|M| \le n/2$, so $|C'| - \tau \le n - \tau \le \delta n \le (\frac{1}{2} - \delta)n \le \tau - |M|$. □

Theorem 6.5 (Conditional consequence)**.** *Assume that for every fixed $\varepsilon_0 > 0$ it is NP-hard to approximate Minimum Vertex Cover within $2 - \varepsilon_0$. If Conjecture 6.1 is proved, then* P = NP*. The assumption holds if UGC holds [11], and in particular if the theorem reported in [12] holds; without any assumption, a proof of Conjecture 6.1 would refute UGC unless* P = NP*.*

Proof. Algorithm 1 runs in polynomial time (Theorem 7.2), and the function $G \mapsto V \setminus S$ returns a vertex cover. A proof of Conjecture 6.1 makes it a $\frac{3}{2}$-approximation (Proposition 6.2), a ratio of the form $2 - \varepsilon_0$ with $\varepsilon_0 = \frac{1}{2}$, so under the assumption it solves an NP-hard problem in polynomial time. The theorem of Khot and Regev [11] derives the assumption from UGC. □

Theorem 6.5 is a conditional implication, not a proof of P = NP. The theorem of [12] had not been peer reviewed at the time of writing; we cite it as reported and have not checked the formalization. By Proposition 6.4, a counterexample to Conjecture 6.1 with $\alpha \le n/4$ does not exist, so a counterexample has an independent set of more than a quarter of the vertices.

Conjecture 6.6 (Ratio $O(\sqrt{n})$)**.** *For every graph G on n vertices without self-loops, the set S returned by Algorithm 1 satisfies $\alpha(G)/|S| \le 2\sqrt{n}$.*

By Theorem 5.4, Conjecture 6.6 holds on every graph that satisfies one of the conditions (a)–(d), in particular on every graph with $\Delta \le \sqrt{n}$; a counterexample has $\Delta > \sqrt{n}$ and satisfies none of (a)–(c).

Theorem 6.7 (Conditional consequence of Conjecture 6.6)**.** *If Conjecture 6.6 is proved, then* P = NP*.*

Proof. Algorithm 1 runs in polynomial time (Theorem 7.2). For $n \ge 16$, $2\sqrt{n} \le n^{3/4}$, so Conjecture 6.6 would give a polynomial-time approximation of Maximum Independent Set within $n^{1-\varepsilon}$ with $\varepsilon = \frac{1}{4}$, which is NP-hard [3]. □

7. Runtime Analysis

Hash-table operations on vertex sets are counted as $O(1)$ expected time, as in the implementation.

Lemma 7.1. *Algorithm 4 runs in $O(B(n + m) + n + m)$ time.*

Proof. The initialisation, the counters and the two counting sorts take $O(n + m + \Delta) = O(n + m)$ time. Every operation of the main loop is charged to the counter work: each pop costs 1, the computation of X costs $\deg u$, every insertion or removal of a vertex v costs $\deg v + 1$, the scan of $N(X)$ costs $\sum_{x \in X} \deg x$, the sort of k refill candidates costs $k \log_2 k$, and the requeueing costs $\sum_{x \in X} \deg x$. The loop runs only while work $\le$ limit $= O(B(n + m) + n + m)$, and a move is started only if work plus the bound $3(T + \deg u + |X| + 1) + T \log_2(T + 1)$ is at most limit. The bound covers every charge of the move except the insertions of refill vertices and their undo, which cost at most $2 \sum_w (\deg w + 1) \le 2(2m + n)$ over distinct vertices w; such a move ends the loop as soon as work exceeds limit. Hence the total work is $O(B(n + m) + n + m)$. □

Theorem 7.2 (Running time)**.** *For fixed budgets B_R, B_L, B_I, Algorithm 1 runs in $O((n + m)\,\alpha_A(n))$ expected time, where α_A is the inverse Ackermann function; in general it runs in $O((n + m)(\alpha_A(n) + B_R + B_L + B_I))$ time.*

Proof. Copying the graph and removing self-loops and isolated vertices take $O(n+m)$ time. Algorithm 2 takes $O(n+m)$ time: every vertex is removed once, every degree decrement costs $O(1)$ and appends one bucket entry, and the pointer moves forward at most Δ times plus once for every backward move, each of which follows a decrement. Salvador takes $O((n+m)\,\alpha_A(n))$ time, and RefineCover and ImproveVertexCover take $O(B(n+m)+n+m)$ time each (Lemma 4.3). Building the adjacency lists, Algorithm 4 (Lemma 7.1), Algorithm 3, the set differences and the comparisons take $O(B(n+m)+n+m)$ time. □

With the default budgets $B_R = B_L = B_I = 100$, the bound of Theorem 7.2 is $O((n+m)\,\alpha_A(n))$. Passing `restart_budget=None` removes the cap of RefineCover, whose running time is then $O(|C|(n+m))$.

8. Experiments

All experiments use Esperanza 0.1.6 and Salvador 0.1.4, and the three scripts of the folder `car/` of the repository: `experiment.py` (Section 8.1), `ugc_candidate.py` (Section 8.2) and `bounded_degree.py` (Section 8.3). The scripts import the repository's own `esperanza/`. Exact optima and bounds are computed with `scipy.optimize.milp` [20], which calls the HiGHS solver [21]. The environment is Windows 11, Python 3.12.3 and NetworkX 3.6.1 [19]; wall-clock times are machine dependent.

8.1. Certified suite of 30 000 graphs

Instance generation. A single master generator `random.Random(42)` is consumed in the order of Table 2. Each random graph receives its own seed, drawn as `randrange(`2^{31}`)` from the master generator. The `amplified` family is built after the first six. Its bases are the connected `gnp` and `regular` instances with $n \le 12$, ranked by their ratio ρ with a stable sort, of which the first 25 are kept; since $\rho = 1$ on all of them, these are the first 25 such instances in generation order. Each amplified instance draws a base uniformly. With probability $1/2$ it is a disjoint union of $c \sim U\{2,\dots,8\}$ copies of the base, and otherwise an independent blow-up with cluster size $t \sim U\{2,\dots,6\}$, in which each vertex becomes t pairwise non-adjacent copies and each edge a complete bipartite join between the corresponding clusters.

Table 2: Instance families of the certified suite (master seed 42). $U\{a,\dots,b\}$ is uniform on integers and $U(a,b)$ uniform on reals; all generators are NetworkX functions. The last two columns give the realised ranges of n and m.

Family	Count	Generator	Parameters	n	m
`gnp`	15000	`gnp_random_graph`(n,p)	$n \sim U\{6,\dots,16\}$, $p \sim U(0.15,0.85)$	6–16	0–109
`bipartite_gnp`	6000	`bipartite.random_graph`(a,b,p)	$a,b \sim U\{3,\dots,14\}$, $p \sim U(0.10,0.50)$	6–28	0–104
`regular`	3000	`random_regular_graph`(d,n)	$d \sim U\{3,\dots,5\}$, $n \in \{\max(d+1,8),\dots,16\}$, nd even	8–16	12–40
`star`	2000	`star_graph`(k)	$k \sim U\{3,\dots,80\}$	4–81	3–80
`complete`	1000	`complete_graph`(n)	$n \sim U\{3,\dots,30\}$	3–30	3–435
`complete_bipartite`	1000	`complete_bipartite_graph`(a,b)	$a \sim U\{2,\dots,12\}$, $b \sim U\{a,\dots,20\}$	4–32	4–240
`amplified`	2000	disjoint union or blow-up	see text	12–96	16–1368

Exact optimum. For each instance we solve

$$\max \sum_{v \in V} x_v \quad \text{subject to} \quad x_u + x_v \le 1 \;\; \forall \{u,v\} \in E, \qquad x_v \in \{0,1\},$$

with the default options of `scipy.optimize.milp` (no time limit; HiGHS relative gap tolerance 10^{-4}), accept a solve only when HiGHS reports an optimal status, and round the objective to the nearest integer; since $\alpha(G) \le 96$, the tolerance is below one vertex, so the rounded value is $\alpha(G)$. Every returned set is verified to be independent. The reported ratio is $\rho = \alpha(G)/|S|$.

Results. Esperanza 0.1.6 returns a maximum independent set on all 30 000 instances; the worst and the mean ratio are 1.000000 in every family (Table 3), and the returned sets have 269 247 vertices in total. The complete run, including the integer programs, took 747.9 s.

Table 3: Certified suite: worst and mean ratio $\alpha(G)/|S|$ of Esperanza 0.1.6 per family.

Family	Count	Worst	Mean
`gnp`	15000	1.000	1.000
`bipartite_gnp`	6000	1.000	1.000
`regular`	3000	1.000	1.000
`star`	2000	1.000	1.000
`complete`	1000	1.000	1.000
`complete_bipartite`	1000	1.000	1.000
`amplified`	2000	1.000	1.000
All	30000	1.000	1.000

8.2. YES-type instances of the Khot–Regev reduction

The instances are those of Part D of the Salvador experiments [16], with the same generator, seed (20261009) and instance names. A Unique Games instance with labels $[k]$ is generated on a random d-regular graph with N vertices and a planted labelling L, each edge carrying a permutation π_{vw} with $\pi_{vw}(L(w)) = L(v)$ (random, or a shift modulo k in the linear variant), and a fraction η of the edges receive a random constraint. Every vertex v becomes a block of s subsets $x \subseteq [k]$ drawn from the p-biased measure, and (v, x) and (w, y) are joined when x and $\pi_{vw}(y)$ are disjoint [11]. The dictator set $\{(v, x) : L(v) \in x\}$, made independent greedily on violated constraints, gives the *planted cover*, an upper bound on τ. The base family has $k = 4$, $p = 0.45$, $\eta = 0$, $d = 4$, $s = 16$, and nine more change one parameter. With $N \in \{40, 80\}$ and two instances per family and size, this gives 40 instances with 640 to 2 560 vertices.

For every instance, L is the larger of the linear relaxation and the dual bound of an integer program (HiGHS, 90 s), and U is the smaller of the planted cover, the integer-program value and the two covers computed; the integer program proved τ on 26 instances. Each instance is classified by Proposition 6.3 as *certified* ($|C| \le 2L - |M|$), *violated* ($|C| > 2U - |M|$) or *open*, for $C = V \setminus S$ (E) and for the Salvador cover (S). Table 4 reports the counts.

Table 4: Khot–Regev YES instances: classification of $|C| \le 2\tau - |M|$ for Esperanza (E) and Salvador (S); "E<S" counts covers of Esperanza strictly smaller than the Salvador cover; "Opt." counts optimal covers of Esperanza; "$\frac{3}{2}$" counts covers with $|C| \le \frac{3}{2}L$; the last two columns give the largest $|C|/L$ and the smallest $|S|/n$.

Family	Inst.	E cert.	E open	E viol.	S cert.	S open	S viol.	E<S	Opt.	$\frac{3}{2}$	max $\lvert C\rvert/L$	min $\lvert S\rvert/n$
Base family ($k = 4$, $p = 0.45$, $\eta = 0$)	4	4	0	0	4	0	0	0	2	4	1.0487	0.4445
Label size $k = 3$	4	4	0	0	4	0	0	0	3	4	1.0232	0.4469
Label size $k = 6$	4	4	0	0	4	0	0	0	4	4	1.0000	0.5055
Bias $p = 0.35$	4	3	1	0	3	1	0	1	1	4	1.1513	0.3812
Bias $p = 0.49$	4	4	0	0	4	0	0	0	4	4	1.0000	0.5070
Noise $\eta = 0.02$	4	4	0	0	4	0	0	0	1	4	1.0468	0.4406
Noise $\eta = 0.1$	4	4	0	0	4	0	0	0	0	4	1.0239	0.4469
Constraint degree $d = 8$	4	4	0	0	4	0	0	0	3	4	1.0067	0.4164
Samples per block $s = 32$	4	3	1	0	3	1	0	0	0	4	1.1483	0.4191
Linear UG, Max-2-Lin($\mathbb{Z}_k$)	4	4	0	0	4	0	0	1	1	4	1.0148	0.4656
All	**40**	**38**	**2**	**0**	**38**	**2**	**0**	**2**	**19**	**40**	**1.1513**	**0.3812**

The inequality $|C| \le 2\tau - |M|$ is certified on 38 instances and violated on none, for both covers. The two open instances have τ unknown: on `kr_p0.35_N80_r0` ($n = 1\,280$), $|C| = 776$, $|M| = 574$ and $L = 674$, so $2L - |M| = 774$; on `kr_s32_N80_r0` ($n = 2\,560$), $|C| = 1\,487$, $|M| = 1\,134$ and $L = 1\,295$, so $2L - |M| = 1\,456$. On both, the two covers have the same size. Esperanza is strictly smaller than Salvador on two instances (683 against 707 on `kr_linear_N80_r0`, and 763 against 766 on `kr_p0.35_N80_r1`). Where τ is known, the cover is optimal on 19 of 26 instances, with mean ratio 1.0024 and maximum 1.0232 (705 against 689 on `kr_s32_N40_r0`, planted cover 690). The cover is no larger than the planted one on 34 instances and strictly smaller on 28. Every instance passes the $\frac{3}{2}$ certificate, and the largest certified ratio is $|C|/L = 1.1513$. In the independent-set form of the reduction, the returned set contains at least $0.3812\,n$ vertices on every instance. Esperanza took at most 5.72 s per instance (mean 2.01 s), and the whole run 2 058.9 s, mostly in the integer programs.

8.3. *Bounded-degree graphs*

The script `bounded_degree.py` measures the ratio on graphs of bounded degree. For every instance it computes S and the range $\alpha_{\text{low}} \le \alpha(G) \le \alpha_{\text{up}}$, where α_{low} is the largest of $|S|$, the planted independent set, the two baselines below and the integer-program value, and α_{up} is the smaller of n minus the linear relaxation of vertex cover and the integer-program dual bound (HiGHS, 60 s). It reports $\alpha_{\text{up}}/|S|$, a certified upper bound on the ratio, the same bound for the Caro–Wei baseline I_{CW} and for the complement of the Salvador cover, and the normalised value $(\alpha_{\text{up}}/|S|)/(\Delta/\log_2^2\Delta)$, with Δ the maximum degree of the instance. There are three families: random Δ-regular graphs; graphs of maximum degree at most Δ with a planted independent set of βn vertices, $\beta \in \{0.35, 0.45\}$, in which every planted vertex has Δ random neighbours outside the planted set and the remaining stubs outside it are paired at random; and Khot–Regev YES graphs with $k = 4$, $p = 0.45$, $\eta = 0$, $d \in \{3, 4\}$ and $s \in \{4, 8, 16\}$ samples per block. The run uses $n = 256$ for the first two families, $N \in \{16, 64\}$ for the third (64 to 1 024 vertices) and one instance per setting, 31 instances in total; the integer program proved α on 17 of them (Table 5).

Table 5: Bounded-degree graphs ($n = 256$ for `regular` and `planted`). "Δ set" is the degree of the setting (d s for `kr`), Δ the largest measured maximum degree; "Exact" counts instances with α proved and "Opt." those on which S is optimal; "up" is $\alpha_{\text{up}}/|S|$ and "low" is $\alpha_{\text{low}}/|S|$; "CW" and "Salv." are $\alpha_{\text{up}}/|I|$ for the Caro–Wei baseline and the complement of the Salvador cover; "norm." is the largest $(\alpha_{\text{up}}/|S|)/(\Delta/\log_2^2\Delta)$.

Family	Δ set	Δ	Inst.	Exact	Opt.	max up	mean up	max low	CW	Salv.	$\Delta/\log_2^2\Delta$	norm.
`kr`	12	12	2	2	2	1.0000	1.0000	1.0000	1.0134	1.0000	0.9337	1.0710
`kr`	24	24	2	2	2	1.0000	1.0000	1.0000	1.0714	1.0000	1.1417	0.8759
`kr`	32	32	2	2	2	1.0000	1.0000	1.0000	1.0246	1.0000	1.2800	0.7812
`kr`	64	64	2	2	1	1.0086	1.0043	1.0086	1.0636	1.0086	1.7778	0.5673
`planted`	4	4	2	1	1	1.0577	1.0288	1.0000	1.1500	1.0577	1.0000	1.0577
`planted`	8	8	2	1	1	1.0769	1.0385	1.0000	1.1136	1.0769	0.8889	1.2115
`planted`	16	16	2	2	2	1.0000	1.0000	1.0000	1.6667	1.0000	1.0000	1.0000
`planted`	32	32	2	2	2	1.0000	1.0000	1.0000	2.3077	1.0000	1.2800	0.7812
`planted`	64	62	2	2	2	1.0000	1.0000	1.0000	1.0000	1.0000	1.7488	0.5815
`regular`	3	3	1	1	0	1.0088	1.0088	1.0088	1.0270	1.0088	1.1942	0.8448
`regular`	4	4	1	0	0	1.0476	1.0476	1.0000	1.1224	1.0476	1.0000	1.0476
`regular`	5	5	1	0	0	1.0947	1.0947	1.0000	1.1556	1.0947	0.9274	1.1804
`regular`	6	6	1	0	0	1.1705	1.1705	1.0000	1.1839	1.1705	0.8979	1.3035
`regular`	8	8	1	0	0	1.2436	1.2436	1.0000	1.3288	1.2436	0.8889	1.3990
`regular`	12	12	1	0	0	1.4194	1.4194	1.0000	1.4915	1.4194	0.9337	1.5201
`regular`	16	16	1	0	0	1.5283	1.5283	1.0000	1.5882	1.5283	1.0000	1.5283
`regular`	24	24	1	0	0	1.6429	1.6429	1.0000	1.7692	1.6429	1.1417	1.4390
`regular`	32	32	1	0	0	1.8788	1.8788	1.0000	2.0667	1.8788	1.2800	1.4678
`regular`	48	48	1	0	0	2.0800	2.0800	1.0000	2.3636	2.0800	1.5389	1.3516
`regular`	64	64	1	0	0	2.2500	2.2500	1.0000	2.5000	2.2500	1.7778	1.2656
`regular`	96	96	1	0	0	2.4286	2.4286	1.0000	2.8333	2.4286	2.2139	1.0969
`regular`	128	128	1	0	0	2.4545	2.4545	1.0000	3.3750	2.4545	2.6122	0.9396

On the `kr` and `planted` families, α is proved on 16 of 18 instances and S is optimal on 15 of them; the largest certified ratio is 1.0769. On the `regular` family, α is proved only for $\Delta = 3$ ($|S| = 113$, $\alpha = 114$), and for $\Delta \ge 4$ the bound α_{up} is the dual bound of the integer program at the time limit; for $\Delta = 128$, $|S| = 11$ and $\alpha_{\text{up}} = 27$. On these instances $\alpha_{\text{low}} = |S|$, so the ratio is only known to lie between 1 and the reported upper bound. On all 31 instances, the set S has the same size as the complement of the Salvador cover. Esperanza took at most 1.48 s per instance, and the whole run 908.8 s.

9. Conclusion

Esperanza 0.1.6 returns a maximal independent set (Theorem 4.5) in $O((n+m)\,\alpha_{\text{A}}(n))$ expected time for fixed work budgets (Theorem 7.2). Its complementary vertex cover has at most $2\tau - r$ vertices (Theorem 5.1), and on graphs of maximum degree Δ its ratio is at most $(\Delta+2)/3$ (Theorem 5.3), above the UGC-hardness threshold $O(\Delta/\log^2\Delta)$ of [9]; its ratio is at most $2\sqrt{n}$ on every graph with $\Delta \le \sqrt{n}$ and on the further classes of Theorem 5.4. On 30 000 graphs with a certified optimum it is optimal on every instance. On 40 Khot–Regev YES instances with 640 to 2 560 vertices, the inequality $|V \setminus S| \le 2\tau - |M|$ is certified on 38 and open on 2, and on 31 bounded-degree graphs the certified upper

bound on the ratio is at most 1.0769 on the families with a planted independent set and at most 2.4545 on random regular graphs of degree up to 128.

We do not claim that Esperanza has ratio $O(\sqrt{n})$ on every graph; this is Conjecture 6.6, and a proof would imply P = NP (Theorem 6.7). Nor do we claim that Esperanza is a $\frac{3}{2}$-approximation for Minimum Vertex Cover. The bound $|V \setminus S| \le 2\tau - |M|$ is a conjecture (Conjecture 6.1); the experiments concern graphs of at most 2 560 vertices; and a proof would refute UGC unless P = NP, and would imply P = NP if the theorem reported in [12] holds (Theorem 6.5). What is guaranteed on a given graph is the per-instance certificate of Proposition 6.3. The implementation is freely available as the `esperanza` package on PyPI [13].

Acknowledgements

The author would like to thank Iris, Marilin, Sonia, Yoselin, and Arelis for their support.

Statements and Declarations

Declaration of interests

The author declares no competing interests.

Funding sources

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Declaration of generative AI in scientific writing

The author designed the strategies of Esperanza, and used Claude Opus 5.5 to implement them by vibe coding and to improve the readability of this manuscript. After using this tool, the author reviewed and edited the code and the content as needed and takes full responsibility for the content of the publication.

Data availability

Code and data: `https://github.com/frankvegadelgado/esperanza` (release `v0.1.6`); the experiments of Section 8 are reproduced by running the scripts of `car/`.

References

[1] R. M. Karp, Reducibility among Combinatorial Problems, in: R. E. Miller, J. W. Thatcher, J. D. Bohlinger (Eds.), Complexity of Computer Computations, Plenum, New York, USA, 1972, pp. 85–103. doi:10.1007/978-1-4684-2001-2_9.

[2] J. Håstad, Clique is hard to approximate within $n^{1-\epsilon}$, Acta Mathematica 182 (1) (1999) 105–142. doi:10.1007/BF02392825.

[3] D. Zuckerman, Linear degree extractors and the inapproximability of Max Clique and Chromatic Number, Theory of Computing 3 (6) (2007) 103–128. doi:10.4086/toc.2007.v003a006.

[4] R. Boppana, M. M. Halldórsson, Approximating maximum independent sets by excluding subgraphs, BIT Numerical Mathematics 32 (2) (1992) 180–196. doi:10.1007/BF01994876.

[5] U. Feige, Approximating maximum clique by removing subgraphs, SIAM Journal on Discrete Mathematics 18 (2) (2004) 219–225. doi:10.1137/S089548010240415X.

[6] M. M. Halldórsson, J. Radhakrishnan, Greed is good: Approximating independent sets in sparse and bounded-degree graphs, Algorithmica 18 (1) (1997) 145–163. doi:10.1007/BF02523693.

[7] E. Halperin, Improved Approximation Algorithms for the Vertex Cover Problem in Graphs and Hypergraphs, SIAM Journal on Computing 31 (5) (2002) 1608–1623. doi:10.1137/S0097539700381097.

[8] S. Khot, On the Power of Unique 2-Prover 1-Round Games, in: Proceedings of the 34th Annual ACM Symposium on Theory of Computing, Association for Computing Machinery, Montreal, Québec, Canada, 2002, pp. 767–775. doi:10.1145/509907.510017.

[9] P. Austrin, S. Khot, M. Safra, Inapproximability of Vertex Cover and Independent Set in Bounded Degree Graphs, Theory of Computing 7 (3) (2011) 27–43. doi:10.4086/toc.2011.v007a003.

[10] S. O. Chan, Approximation Resistance from Pairwise-Independent Subgroups, Journal of the ACM 63 (3) (2016) 27:1–27:32.

[11] S. Khot, O. Regev, Vertex Cover Might Be Hard to Approximate to Within $2-\epsilon$, Journal of Computer and System Sciences 74 (3) (2008) 335–349. doi:10.1016/j.jcss.2007.06.019.

[12] OpenAI Math Team, The Unique Games Conjecture and Optimal Approximation Thresholds (Result Family 102), GitHub Repository, manuscripts and Lean artifacts available at https://github.com/openai/math/blob/main/lean/docs/102.md (October 2026). URL https://github.com/openai/math
[13] F. Vega, Esperanza: Approximate Independent Set Solver, https://pypi.org/project/esperanza, python package version 0.1.6 (2026).
[14] Y. Caro, New results on the independence number, Tech. rep., Tel-Aviv University (1979).
[15] V. K. Wei, A lower bound on the stability number of a simple graph, Tech. Rep. Technical Memorandum 81-11217-9, Bell Laboratories (1981).
[16] F. Vega, An Approximate Solution to the Minimum Vertex Cover Problem: The Salvador Algorithm, manuscript; preprint at https://www.preprints.org/manuscript/202605.2000 (2026).
[17] F. Vega, Salvador: Approximate Vertex Cover Solver, https://pypi.org/project/salvador, python package version 0.1.4 (2026).
[18] D. V. Andrade, M. G. C. Resende, R. F. Werneck, Fast local search for the maximum independent set problem, Journal of Heuristics 18 (4) (2012) 525–547. doi:10.1007/s10732-012-9196-4.
[19] A. A. Hagberg, D. A. Schult, P. J. Swart, Exploring network structure, dynamics, and function using NetworkX, in: Proceedings of the 7th Python in Science Conference (SciPy 2008), Pasadena, CA, USA, 2008, pp. 11–15.
[20] P. Virtanen, R. Gommers, T. E. Oliphant, et al., SciPy 1.0: Fundamental algorithms for scientific computing in Python, Nature Methods 17 (2020) 261–272. doi:10.1038/s41592-019-0686-2.
[21] Q. Huangfu, J. A. J. Hall, Parallelizing the dual revised simplex method, Mathematical Programming Computation 10 (1) (2018) 119–142. doi:10.1007/s12532-017-0130-5.