Algorithms of the Domination Lab

Davood Bakhshesh
University of Bojnord

Version 2026.11

Notation

Throughout, $G = (V, E)$ is a simple undirected graph with $n = |V|$ vertices and $m = |E|$ edges. $N(v)$ is the set of neighbors of v, $N[v] = N(v) \cup \{v\}$, and for a set X we write $N[X] = \bigcup_{v \in X} N[v]$. Δ and δ are the maximum and minimum degree, and $H(k) = \sum_{i=1}^{k} 1/i \le \ln k + 1$ is the harmonic number. The parameters computed by the lab are γ (domination), γ_t (total), γ_c (connected), i (independent domination), γ_{pr} (paired), $\gamma_{\times 2}$ (2-domination), γ_p (perfect), α (independence), β (vertex cover), γ_R (Roman) and d (domatic number); see [11]. These problems are NP-hard on general graphs (for the classical ones see [8]), which is why the lab uses heuristics for large graphs and exhaustive search for small ones.

1 Greedy dominating set

A vertex v covers the set $N[v]$, so finding a dominating set is a set cover problem on the universe V. The greedy rule repeatedly takes the vertex that covers the largest number of vertices that are not covered yet.

Analysis

- Correctness. While some vertex is uncovered, that vertex itself has gain at least 1, so every round covers a new vertex and the loop ends with $C = V$, i.e. S is dominating.
- Time. At most n rounds, each scanning all vertices and their neighbors: $O(n(n + m))$.
- Guarantee. This is the greedy algorithm for set cover with sets $N[v]$ of size at most $\Delta + 1$, hence $|S| \le H(\Delta + 1)\,\gamma(G) \le (\ln(\Delta + 1) + 1)\gamma(G)$ [14, 15, 1].

2 Greedy total dominating set

In a total dominating set every vertex, including those in S, needs a neighbor in S. A vertex therefore covers only its open neighborhood $N(v)$, and the greedy idea is again set cover. If some vertex has no neighbor, no total dominating set exists [13].

Analysis

- Correctness. A vertex u is covered by v exactly when $u \in N(v)$, so $C = V$ means that every vertex has a neighbor in S.
- Time. $O(n(n + m))$.
- Guarantee. Set cover with sets of size at most Δ, so $|S| \le H(\Delta)\,\gamma_t(G) \le (\ln \Delta + 1)\gamma_t(G)$ [1].

Algorithm 1 Greedy-Dominating-Set

Require: graph $G = (V, E)$
Ensure: a dominating set S
$S \leftarrow \emptyset$; $C \leftarrow \emptyset$ ▷ C = covered vertices
while $C \neq V$ **do**
    pick v maximizing $|N[v] \setminus C|$
    $S \leftarrow S \cup \{v\}$; $C \leftarrow C \cup N[v]$
end while
return S

Algorithm 2 Greedy-Total-Dominating-Set

Require: graph G
Ensure: a total dominating set, or a message that none exists
$S \leftarrow \emptyset$; $C \leftarrow \emptyset$
while $C \neq V$ **do**
    pick v maximizing $|N(v) \setminus C|$
    if $|N(v) \setminus C| = 0$ **then** ▷ some vertex has no neighbor
        return "no total dominating set"
    end if
    $S \leftarrow S \cup \{v\}$; $C \leftarrow C \cup N(v)$
end while
return S

3 Connected dominating set (greedy growth)

Start from a vertex of maximum degree. At each step add a vertex that is already dominated, i.e. adjacent to S, and that dominates the most new vertices. Because every new vertex touches S, the induced subgraph $G[S]$ stays connected. If no such vertex exists before everything is dominated, G is disconnected. Guha and Khuller [9] analyse this kind of greedy growth: they show that the naive rule can perform badly, and that a variant which also looks at pairs of vertices (a dominated vertex together with a white neighbor) achieves $2(1 + H(\Delta))$. The lab runs the simple rule, so no approximation ratio is claimed for it.

Analysis

- Correctness. $u \in C \setminus S$ is adjacent to a vertex of S, so $G[S \cup \{u\}]$ is connected whenever $G[S]$ is; the final set is dominating because $C = V$.
- Time. $O(n(n + m))$.

4 Paired dominating set (greedy pairs)

A paired dominating set is a dominating set whose induced subgraph has a perfect matching [12]. Take a vertex u that is not dominated yet, pair it with the neighbor w that covers the most new vertices, and add both. Each pair is an edge, so the chosen pairs form a perfect matching of $G[S]$ by construction. A vertex without neighbors makes the problem unsolvable.

Analysis

- Correctness. $C = V$ at the end, so S dominates, and $G[S]$ contains the disjoint edges uw that cover S.

Algorithm 3 Greedy-CDS

Require: connected graph G
Ensure: a connected dominating set S

$v \leftarrow$ vertex of maximum degree; $S \leftarrow \{v\}$; $C \leftarrow N[v]$
while $C \neq V$ **do**
  pick $u \in C \setminus S$ maximizing $|N(u) \setminus C|$
  if $|N(u) \setminus C| = 0$ **then** ▷ G is disconnected
    return "disconnected"
  end if
  $S \leftarrow S \cup \{u\}$; $C \leftarrow C \cup N[u]$
end while
return S

Algorithm 4 Greedy-Paired-Dominating-Set

Require: graph G
Ensure: a paired dominating set, or a message that none exists

$S \leftarrow \emptyset$; $C \leftarrow \emptyset$
while $C \neq V$ **do**
  $u \leftarrow$ any vertex of $V \setminus C$
  if $N(u) = \emptyset$ **then**
    return "no paired dominating set"
  end if
  $w \leftarrow$ neighbor of u maximizing $|N[w] \setminus C|$
  $S \leftarrow S \cup \{u, w\}$; $C \leftarrow C \cup N[u] \cup N[w]$
end while
return S ▷ the edges uw form a perfect matching of $G[S]$

- Time. $O(n(n+m))$. The result is always valid; no approximation ratio is claimed for this heuristic. For graphs without isolated vertices $\gamma \leq \gamma_{pr} \leq 2\gamma$ [12].

5 2-dominating set (greedy)

In a 2-dominating set every vertex outside S has at least two neighbors in S [6]. For each vertex we count how many neighbors it is still missing, and at each step we add the vertex that removes the most missing neighbors: its own need disappears, and every neighbor outside S that still needs help gets one step closer.

Analysis

- Correctness. The loop stops only when every vertex has def $= 0$, which is exactly the definition. It terminates because adding any vertex with positive gain lowers $\sum \text{def}$ by at least 1.
- Time. Polynomial: at most n rounds, each looking at every vertex and its neighbors. No approximation ratio is claimed.

6 Roman domination from a greedy dominating set

A Roman dominating function is a labeling $f : V \to \{0, 1, 2\}$ such that every vertex with $f(v) = 0$ has a neighbor u with $f(u) = 2$; its weight is $\sum_v f(v)$ and γ_R is the minimum weight

Algorithm 5 Greedy-2-Dominating-Set

Require: graph G
Ensure: a 2-dominating set S

$S \leftarrow \emptyset$ ▷ $\mathrm{def}(v) = 0$ if $v \in S$, else $\max\{0,\, 2 - |N(v) \cap S|\}$
while $\sum_v \mathrm{def}(v) > 0$ **do**
  pick $x \notin S$ maximizing $\mathrm{def}(x) + |\{u \in N(x) \setminus S : \mathrm{def}(u) > 0\}|$
  $S \leftarrow S \cup \{x\}$
end while
return S

Algorithm 6 Roman-From-Greedy-DS

Require: graph G
Ensure: a Roman dominating function f

$S \leftarrow$ Greedy-Dominating-Set(G)
$f(v) \leftarrow 2$ for $v \in S$; $f(v) \leftarrow 0$ otherwise
for $v \in S$ **do** ▷ improvement step
  if every $u \in N(v)$ with $f(u) = 0$ has some $w \in N(u) \setminus \{v\}$ with $f(w) = 2$ **then**
    $f(v) \leftarrow 1$
  end if
end for
return f

[3, 16]. Labeling a dominating set S with 2 and everything else with 0 gives weight $2|S|$. The improvement step lowers a 2 to a 1 when this is safe.

Analysis

- Correctness. Initially f is valid because S is dominating. Lowering $f(v)$ from 2 to 1 is safe: v itself needs no neighbor labeled 2, and each neighbor labeled 0 still has another neighbor labeled 2.
- Time. $O(n(n+m))$. The weight is at most $2|S|$, which is why $\gamma \le \gamma_R \le 2\gamma$ in general [3]. For this heuristic, weight $\le 2H(\Delta+1)\,\gamma_R$, since $|S| \le H(\Delta+1)\gamma \le H(\Delta+1)\gamma_R$.

7 Vertex cover: 2-approximation by a maximal matching

Scan the edges. Whenever an edge has no endpoint in the cover yet, put both endpoints in. The edges that triggered this are pairwise disjoint, so they form a maximal matching M [5].

Analysis

- Correctness. Every edge is either already covered when examined or gets both endpoints added.
- Guarantee. The chosen edges are disjoint, and any vertex cover must contain an endpoint of each of them, so $\beta(G) \ge |M|$ while $|C| = 2|M|$; hence $|C| \le 2\beta(G)$. Time $O(m)$. Also $\alpha + \beta = n$ [7].

8 Greedy maximal independent set (smallest degree first)

Repeatedly take a vertex of smallest degree in what is left of the graph, add it to S, and delete it together with its neighbors. Taking low-degree vertices first deletes few vertices per step, which

Algorithm 7 Matching-Vertex-Cover

Require: graph $G = (V, E)$
Ensure: a vertex cover C
1: $C \leftarrow \emptyset$
2: **for** each edge $uv \in E$ (in input order) **do**
3: **if** $u \notin C$ and $v \notin C$ **then**
4: $C \leftarrow C \cup \{u, v\}$
5: **end if**
6: **end for**
7: **return** C

Algorithm 8 Greedy-Independent-Set

Require: graph G
Ensure: a maximal independent set S
1: $S \leftarrow \emptyset$; $H \leftarrow G$
2: **while** $V(H) \neq \emptyset$ **do**
3: $v \leftarrow$ vertex of minimum degree in H
4: $S \leftarrow S \cup \{v\}$; $H \leftarrow H - N_H[v]$
5: **end while**
6: **return** S

usually leaves room for a larger set.

Analysis

- Correctness. Vertices are removed only together with a neighbor in S or when chosen, so S is independent and no vertex can be added: S is maximal.
- Guarantee. Every maximal independent set has at least $n/(\Delta+1)$ vertices. For this minimum-degree rule Halldórsson and Radhakrishnan prove the better ratio $(\Delta + 2)/3$ on graphs of maximum degree Δ: $\alpha(G) \leq \frac{\Delta+2}{3}|S|$ [10]. Time $O(n(n + m))$.

9 Greedy independent dominating set (largest degree first)

An independent dominating set is exactly a maximal independent set [4]: a maximal independent set is dominating (otherwise another vertex could be added), and a dominating independent set is maximal. To keep the set small, this variant takes a vertex of largest degree first, so that each step deletes as many vertices as possible.

Analysis

- Correctness. As for the previous algorithm, S is a maximal independent set and hence independent and dominating.
- Time $O(n(n + m))$. $|S|$ is an upper bound for $i(G)$, and $\gamma \leq i \leq \alpha$ [4]. No approximation ratio is claimed.

10 Exact answers (bit masks, up to 20 vertices)

Because these problems are NP-hard [8], exact answers take exponential time. For $n \leq 20$ the lab looks at every subset $X \subseteq V$, written as a bit mask. Two tables are filled once by dynamic

Algorithm 9 Greedy-Independent-Dominating-Set

Require: graph G
Ensure: an independent dominating set S

$S \leftarrow \emptyset$; $H \leftarrow G$
while $V(H) \neq \emptyset$ **do**
    $v \leftarrow$ vertex of maximum degree in H
    $S \leftarrow S \cup \{v\}$; $H \leftarrow H - N_H[v]$
end while
return S

Algorithm 10 Exact-Search

Require: graph G with $n \leq 20$, a problem P
Ensure: an optimal set for P, if one exists

for $X = 1, \ldots, 2^n - 1$, with x the lowest vertex of X **do**
    $\mathrm{cl}[X] \leftarrow \mathrm{cl}[X \setminus \{x\}] \cup N[x]$
    $\mathrm{ind}[X] \leftarrow \mathrm{ind}[X \setminus \{x\}] \wedge (N(x) \cap X = \emptyset)$
end for
best $\leftarrow$ none
for $X \subseteq V$ **do**
    if $|X|$ cannot improve $|best|$ **then**
        continue
    end if
    if $\mathrm{Test}_P(X)$ **then**
        best $\leftarrow X$
    end if
end for
return *best*

programming over the masks: the closed neighborhood $N[X]$ of each subset, and whether X is independent. Afterwards each problem is a quick test on a subset, and subsets that cannot beat the best answer found so far are skipped.

Analysis

- Tests. Dominating: $\mathrm{cl}[X] = V$. Total: $\bigcup_{v \in X} N(v) = V$. Connected: dominating and $G[X]$ connected. Independent dominating: $\mathrm{ind}[X]$ and $\mathrm{cl}[X] = V$. Paired: dominating and $G[X]$ has a perfect matching. 2-dominating / perfect: every $v \notin X$ has at least 2 / exactly 1 neighbor in X. Independent set: $\mathrm{ind}[X]$. Vertex cover: $\mathrm{ind}[V \setminus X]$.
- Roman domination. Let X be the set of vertices labeled 2. Every vertex outside $N[X]$ must get label 1, all others 0, so the best weight for this X is $2|X| + |V \setminus N[X]|$, and we minimize it over all X.
- Time $O(2^n \cdot n)$ in the worst case and memory $O(2^n)$.

11 Domatic partition

The domatic number $d(G)$ is the largest number of pairwise disjoint dominating sets; it satisfies $d(G) \leq \delta + 1$ [4]. Every dominating set contains a minimal dominating set, so it is enough to find all minimal dominating sets and to search, by backtracking, for the largest family of pairwise disjoint ones.

Algorithm 11 Domatic-Partition

Require: graph G with $n \le 20$
Ensure: disjoint dominating sets $D_1, \dots, D_k$

$\mathcal{M} \leftarrow$ all minimal dominating sets of G; $best \leftarrow \emptyset$
Search($s,\ U,\ \mathcal{F}$) ▷ backtracking over $\mathcal{M}$
if $|\mathcal{F}| > |best|$ **then**
    $best \leftarrow \mathcal{F}$
end if
if $|best| = \delta + 1$ or the step limit is reached **then**
    return stop
end if
for $D \in \mathcal{M}$ from index s with $D \cap U = \emptyset$ **do**
    Search(next, $U \cup D,\ \mathcal{F} + [D]$)
end for
add vertices that are in no set to D_1
return *best*

Algorithm 12 Check-Selection

Require: graph G, a selection S (or labels f), a problem P
Ensure: the vertices and edges that violate the definition

for $v \in V$ **do**
    $k(v) \leftarrow |N(v) \cap S|$
end for
for $v \in V$ **do**
    compare $k(v)$ with the definition of P ▷ e.g. $v \in S$ or $k(v) \ge 1$
end for
for $uv \in E$ **do**
    check independence / cover conditions on u, v
end for
test connectivity or a perfect matching of $G[S]$ if P requires it

Analysis

- Correctness. Replacing each set of an optimal family by a minimal dominating set contained in it keeps the sets disjoint, so searching only $\mathcal{M}$ loses nothing; adding leftover vertices to a set keeps it dominating.
- Time. Exponential. The search stops after $2 \cdot 10^6$ steps; the lab then reports only a lower bound $d(G) \ge k$.

12 Checking your selection

Whatever you select, the lab counts for every vertex how many of its neighbors are in S (and, for Roman domination, how many have label 2) and compares this with the definition. Connectivity of $G[S]$ is tested with a graph search; the perfect matching of paired domination with a short backtracking search (up to $|S| = 24$).

Analysis

- Time $O(n + m)$ for most problems.

Algorithm 13 Unit-Disk-Graph

Require: points $p_1, \ldots, p_n$ in the plane, radius r
Ensure: the graph $G = (V, E)$

```
E ← ∅
for all pairs i < j do
    if ‖p_i − p_j‖ ≤ r then
        add the edge ij to E
    end if
end for
return (V, E)
```

Algorithm 14 Read-Ipe

Require: an Ipe XML file
Ensure: a graph G

```
V ← all marks and small circles (after group transformations)
for each polyline or polygon and each consecutive pair of points p, q do
    a ← vertex nearest to p within tolerance, else a new vertex at p
    b ← the same for q
    if a ≠ b then
        add the edge ab (once)
    end if
end for
scale and center everything to fit the canvas
```

13 Unit disk graphs

In unit disk mode the lab rebuilds the edges every time a vertex moves: two vertices are adjacent when their Euclidean distance is at most the radius r [2].

Analysis

- Time $O(n^2)$ per update.

14 Reading Ipe files

Ipe [17] stores a drawing as XML. The lab turns marks and small circles into vertices, and every segment of a polyline or polygon into an edge between the vertices nearest to its two endpoints (within a small tolerance). An endpoint with no vertex nearby becomes a new vertex. Transformations of groups are applied, and the result is scaled to fit the canvas.

Analysis

- Time $O(n \cdot k)$ in the worst case, where k is the number of segments.

References

[1] Chvátal, V.: A greedy heuristic for the set-covering problem. Mathematics of Operations Research 4(3), 233–235 (1979).

[2] Clark, B.N., Colbourn, C.J., Johnson, D.S.: Unit disk graphs. Discrete Mathematics 86(1–3), 165–177 (1990).

[3] Cockayne, E.J., Dreyer Jr., P.A., Hedetniemi, S.M., Hedetniemi, S.T.: Roman domination in graphs. Discrete Mathematics 278(1–3), 11–22 (2004).

[4] Cockayne, E.J., Hedetniemi, S.T.: Towards a theory of domination in graphs. Networks 7(3), 247–261 (1977).

[5] Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 4th edn. MIT Press (2022).

[6] Fink, J.F., Jacobson, M.S.: n-domination in graphs. In: Graph Theory with Applications to Algorithms and Computer Science, pp. 283–300. Wiley, New York (1985).

[7] Gallai, T.: Über extreme Punkt- und Kantenmengen. Annales Universitatis Scientiarum Budapestinensis, Sectio Mathematica 2, 133–138 (1959).

[8] Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman, San Francisco (1979).

[9] Guha, S., Khuller, S.: Approximation algorithms for connected dominating sets. Algorithmica 20(4), 374–387 (1998) doi:https://doi.org/10.1007/PL00009201.

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

[11] Haynes, T.W., Hedetniemi, S.T., Slater, P.J.: Fundamentals of Domination in Graphs. Marcel Dekker, New York (1998).

[12] Haynes, T.W., Slater, P.J.: Paired-domination in graphs. Networks 32(3), 199–206 (1998).

[13] Henning, M.A., Yeo, A.: Total Domination in Graphs. Springer Monographs in Mathematics, Springer (2013).

[14] Johnson, D.S.: Approximation algorithms for combinatorial problems. Journal of Computer and System Sciences 9(3), 256–278 (1974).

[15] Lovász, L.: On the ratio of optimal integral and fractional covers. Discrete Mathematics 13(4), 383–390 (1975).

[16] Stewart, I.: Defend the Roman Empire! Scientific American 281(6), 136–139 (1999).

[17] Cheong, O.: Ipe, the extensible drawing editor. https://ipe.otfried.org.