Quality Certification of Heuristics on Real-World Graphs

Fabrice Lécuyer
fabrice.lecuyer@lip6.fr
Sorbonne Université, CNRS, LIP6
F-75005 Paris, France

ABSTRACT

Web platforms receive unprecedented amounts of queries and must respond in a heartbeat for user comfort. Yet, some crucial algorithmic problems are hard to solve on big instances. Part of them, such as the vertex cover problem, have a k-approximation algorithm: the result is guaranteed to be within a factor k of the optimum one. But even then, real-world applications obtain better results with fast algorithms that have no theoretical guarantees, called heuristics. There is however no indication on how far these heuristic results are from optimum. To address this issue, we propose a method to certify the quality of a heuristic on a given instance. The quality certification consists in comparing the experimental result to a bound of the optimum value, obtained through a different heuristic. We show two ways of obtaining bounds and illustrate them with famous graph problems: vertex cover for bounds that derive from a k-approximation algorithm, and independent set for bounds given by a complementary problem. Tested on 114 real-world networks with up to three billion edges, our method certifies that the results of state-of-the-art heuristics for both problems are within 10% of the optimum value on more than half of the networks. This work shows that valuable quality certificates can be given for existing heuristics on specific instances without loosing on scalability. As it generalises to algorithmic problems with a k-approximation, it opens a door for further research and for deployment in real-world applications.

KEYWORDS

graph algorithm, real-world network, scalability, approximation, vertex cover, independent set

1 INTRODUCTION

The curse of applied algorithmics is that many of the most crucial problems are NP-hard, which generally means that exact algorithms cannot scale to massive datasets. Online applications such as web browsers or social platforms are particularly sensitive to this problem, as they often need to respond with minimum latency to an unforeseen user query. This leads to a split between exact methods that push the scalability further and further, and quick methods that get closer and closer to optimum results. The latter is particularly efficient on real-world instances such as social networks, web graphs or biological networks, which rarely resemble worst-case scenarios and allow heuristics to give excellent results [17].

However, while heuristic results can be compared to one another, it is not possible to measure how far they are from optimal. This work introduces a method to certify the quality of a result on a given instance. It combines a heuristic for the initial problem with another heuristic that gives an instance-specific bound of the optimal solution. Together, the approximate result and the bound allow us to compute a more precise approximation quality. We call this approach a *quality certification*. We showcase it with two famous graph problems: minimum vertex cover and maximum independent set, that have applications in network robustness [31], wireless communication [33], virus transmission [30] and image rendering [10].

These two problems are known to be theoretically hard, as they both figure in the 21 NP-complete problems described in 1972 by Karp[1]. They are complementary to each other, since the nodes that are not in a vertex cover form an independent set. Vertex cover has a 2-approximation algorithm that consists in finding any maximal matching, but it is thought that no lower constant ratio can be achieved [6, 13, 25]. As for independent set, it is hard to approximate within any constant factor on general graphs. Tighter approximation ratios exist for graphs that have typical properties of real-world networks, such as bounded degrees [5, 22], high clustering [8], or a power-law degree distribution [19]. Reduction and kernelisation rules, a survey of which is given in [18], diminish the instance size and can lead to better approximation guarantees [3]. The two problems have been generalised to weighted graphs [11, 32], dynamic networks [1, 4] or partial coverage [24], and analysed in the quadratic programming [28] and massively parallel settings [20].

In practice, there is a trade-off between speed and guarantee of quality. On the one hand, the quality ratio of fast heuristics [12] and 2-approximation algorithms for vertex cover is usually way below 2 when compared to an optimum solution [21]. Yet, in the more interesting case when the optimum is unknown, it is not possible to measure how accurate the result is. On the other hand, exponential algorithms for exact solutions can be extremely fast on real-world networks: the 2019 PACE challenge [15] fostered efforts towards quick and exact algorithms for vertex cover, and its laureate [23] solves some graphs of millions of nodes in a few seconds. Still, our experiments show that it fails to solve the problem in reasonable time for larger or more complex graphs.

A quality certification bridges this gap: it takes a dataset and gives both an approximate result and a certificate of its quality, defined as the ratio between the heuristic result and a bound on the optimum value. For example, the shortest path between two cities is lower-bounded by the distance as the crow flies; the certified quality of a path is then given by the ratio between its length and the lower-bound. However, such bounds are hard to obtain in general. The key insight of this paper is that multiple problems of interest can be bounded empirically, using heuristics to obtain a high lower-bound – or a low upper-bound – on a specific instance. We propose two distinct bounding ideas for important graph problems: for vertex cover, we use the 2-approximation provided by a maximum matching; for independent set, we reverse the bound of vertex cover, which is the complementary problem.

[1]The list contains Node Cover (another name for Vertex Cover), and Set Packing which is equivalent to Independent Set and shares its approximation properties.

The rest of the paper is organised as follows. Section 2 presents the mathematical formalism with notations and definitions of graph problems. Section 3 details the quality certification method and its specifics in the case of vertex cover and independent set. Section 4 presents experiments on more than a hundred networks that show the relevance and scalability of the method, and discusses possible improvements.

2 BACKGROUND

2.1 Notations

We consider an unweighted undirected simple graph $G = (V, E)$ with $n = |V|$ vertices and $m = |E|$ edges. The set of neighbours of a vertex u is denoted $N_u = \{v \mid \{u, v\} \in E\}$, and its degree is $d_u = |N_u|$. For an edge $e = \{u, v\}$, we say that u and v are the extremities of e and that e is adjacent to u and v. The adjacent edges of a subset $W \subseteq V$ are all the edges with at least one extremity in W. Two edges are adjacent when they share an extremity. For graph problems, a solution is *optimal* (or minimal or maximal) when it cannot be modified into a better solution by adding or removing an element, and *optimum* (or minimum or maximum) if it is as good as any other solution.

2.2 Vertex Cover

Definition 2.1 (Vertex cover, minimal, minimum). A vertex cover is a set of nodes C that is adjacent to every edge of the graph: $\forall\{u, v\} \in E,\ u \in C$ or $v \in C$. To simplify, it can be referred to as a *cover*. A vertex cover C is *minimal* if removing any node uncovers an edge: $\forall v \in C,\ C \setminus \{v\}$ is not a cover. It is *minimum* if it is as small as any other vertex cover.

Property 1 (Hardness). *Knowing if there exists a vertex cover of a given size is NP-complete. Thus, finding a minimum vertex cover is NP-hard.*

Property 2 (Approximability). *It is NP-hard to approximate the size of a minimum vertex cover by a constant factor lower than 2 under the unique games conjecture [25]. A 2-approximation is given by any maximal matching as defined further.*

2.3 Independent Set

Definition 2.2 (Independent Set, maximal, maximum). An independent set is a set S of non-adjacent nodes: $\forall v \in V,\ \{u, v\} \in E,\ u \notin S$ or $v \notin S$. The independent set S is *maximal* if each node that is not in the set has a neighbour in it: $\forall v \notin S,\ S \cup \{v\}$ is not an independent set. It is *maximum* if it is as large as any independent set.

Note that an independent set is the complement of a vertex cover: consider a set $C \subseteq V$ and $S = V \setminus C$; if C is a cover then each edge has at least one extremity in C, so at most one in S, which is equivalent to say that S is an independent set as none of its nodes are adjacent.

Property 3 (Hardness). *Because of the above complementarity, knowing if there exists an independent set of a given size is NP-complete and finding a maximum independent set is NP-hard.*

Property 4 (Inapproximability). *It is NP-hard to approximate the size of a maximum independent set by a constant factor.*

Remark. While finding a minimum vertex cover instantly gives a maximum independent set, the approximability of the two problems differs: a 2-approximation for vertex cover is not necessarily a $\frac{1}{2}$-approximation for independent set. For instance, if $|V| = 100$ and the minimum vertex cover has size 40, a vertex cover of size 80 is a factor 2 approximation for vertex cover. But the corresponding independent set is of size 20 for an optimal value of 60, which gives a factor $\frac{1}{3}$ approximation.

2.4 Matching

Definition 2.3 (Matching, maximal, maximum, minimum maximal). A matching M is a set of non-adjacent edges: $\forall e, f \in M,\ e \cap f = \emptyset$. A matching M is *maximal* when it is adjacent to all other edges of the graph: $\forall e \in E, \exists f \in M, e \cap f \neq \emptyset$. A *maximum* matching is a maximal matching that is as large as any other matching. A *minimum maximal* matching is a maximal matching that is as small as any other maximal matching.

Property 5 (Hardness). *A maximum matching can be found in polynomial time. However, it is NP-complete to know if there exists a minimum maximal matching of a given size, and thus NP-hard to find one.*

Definition 2.4 (Cover of a matching). The nodes of any maximal matching M form a minimal vertex cover noted C_M and called the cover of the matching:

$$C_M = \bigcup_{\{u,v\} \in M} \{u\} \cup \{v\} \subseteq V$$

Property 6 (Cover approximation). *For any maximal matching M, the vertex cover C_M is at most twice as large as any cover. In particular, a minimum cover C^* satisfies $\frac{1}{2}|C_M| \leq |C^*| \leq |C_M|$.*

Proof. The edges of M are non-adjacent. For each of them, C_M contains 2 nodes, but C^* must contain at least 1 node in order to cover this edge. □

3 QUALITY CERTIFICATION METHOD

This section develops the general method of quality certification with bounds of the optimum value. It also shows two distinct ways of obtaining bounds: leveraging a 2-approximation algorithm (for vertex cover) or using a complementary problem (for independent set).

3.1 Certification in general

Consider a minimisation problem $\mathcal{P}$ that has an unknown minimum value p^*. Heuristics can be designed to obtain an approximate value $h \geq p^*$, with the goal of being as close to p^* as possible; we call them *solution-heuristics*. Now suppose that another algorithm, that we call *bounding-heuristic*, is able to produce a positive lower-bound $b \leq p^*$ for this problem. It means that the unknown value p^* satisfies $p^* \in [b, h]$. This implies that $\frac{h}{p^*} \leq \frac{h}{b}$ which, by definition, means that the value h is a $\frac{h}{b}$-approximation of the optimal value p^*. The general method is summed up in Algorithm 1.

Proposition 3.1. *If h is an approximate solution to a minimisation problem $\mathcal{P}$ and b is a lower-bound on the minimum p^*, then h is within a factor $\frac{h}{b}$ of optimum.*

Algorithm 1 – Quality certification for a minimisation problem $\mathcal{P}$

Input: instance of $\mathcal{P}$ with unknown minimum p^*
1: with a solution-heuristic, obtain a solution H of value $h \geq p^*$
2: with a bounding-heuristic, obtain a lower-bound $b \leq p^*$
3: **return** solution H and certified quality ratio $\frac{h}{b}$

The goal is then to reduce the interval $[b, h]$ by finding solution-heuristics with low value h and bounding-heuristics with high value b. To find good solution-heuristics, it is possible to look into the existing literature, that provides a variety of methods for the most common graph problems and their variants.

To find good bounding-heuristics, we propose two approaches that may generalise to other algorithmic problems. First, we show how a poor approximation can certify vertex cover: if a solution-heuristic is proven to be a k-approximation, its result h satisfies $h \leq k \cdot p^*$, which also provides a lower bound $b = \frac{h}{k}$. Such an algorithm is normally designed to output a good, low value h, but tweaking it to obtain a poor, high value produces a tighter lower-bound. Second, for problems such as independent set that are hard to approximate, we show a complementary bounding: it consists in translating the bounds of a complementary problem into bounds of the initial problem.

3.2 Certifying vertex cover

This section showcases our quality certification method on the minimum vertex cover problem. We first show why a 2-approximation algorithm is both a solution-heuristic and a bounding-heuristic, using the idea of poor approximations; note that this idea applies to any problem that has a k-approximation even if $k \neq 2$. Second, we present refined bounding-heuristics and solution-heuristics. This allows us to formerly define a quality certificate for vertex cover.

Algorithm 2 – Edge-greedy 2-approximation for vertex cover

Input: priority function $\eta : E \to \mathbb{Z}$
1: start with empty cover: $C_M \leftarrow \emptyset$
2: **while** there are uncovered edges **do**
3: take edge $\{u, v\}$ with highest priority $\eta(\{u, v\})$
4: add u and v to cover: $C_M \leftarrow C_M \cup \{u\} \cup \{v\}$
5: **return** C_M

3.2.1 Greedy vertex cover approximations. The minimum vertex cover problem is NP-complete, but it has a simple 2-approximation algorithm presented as Algorithm 2. It is a linear-time greedy algorithm that repeatedly adds both extremities of an uncovered edge to the cover until all edges are covered. This selection of edges forms a maximal matching M, so per Property 6, the resulting cover C_M has at most twice as many nodes as an optimal cover C^*. Thus, Algorithm 2 serves both as a solution-heuristic and as a bounding-heuristic, and any cover C_M obtained with it satisfies

$$\frac{1}{2}|C_M| \leq |C^*| \leq |C_M|$$

In any case, C_M is a 2-approximation and a typical solution-heuristic will make it as small as possible to be closer to the minimum cover size. Conversely, the poor approximation strategy consists in making it as *large* as possible: this maximises the lower-bound, which implies a larger minimum cover, and hence certifies a better quality ratio.

Interestingly, Algorithm 2 leads to covers of different sizes depending on the priority order η in which edges are considered at line 3. The intuition is that a small cover C_M^- may be obtained by selecting nodes with high degree, because they will cover more edges. On the other hand, selecting nodes with low degree is likely to result in a large cover C_M^+. Finding priority functions for larger C_M^+ and smaller C_M^- tightens the above equation, which becomes:

$$\frac{1}{2}|C_M^+| \leq |C^*| \leq |C_M^-|$$

3.2.2 Bounding-heuristics. To maximise the lower-bound, we need to obtain a poor approximation from Algorithm 2, or in other words, a cover C_M that is as large as possible. As C_M corresponds to a matching M, we are in fact searching for a maximum matching. The maximum matching problem can be solved exactly with the famous blossom algorithm [16] in polynomial time $O(n^2 m)$ for any graph with n nodes and m edges. Other algorithms lower the complexity to $O(\sqrt{n}m)$ [7, 27] and a linear-time approximation scheme exists [14].

Besides, Algorithm 2 runs in time $O(m)$ and can also provide a large matching, with the low-degree first priority function η_{low}. At each step, note x_u the number of neighbours of node u that are not in the cover yet. Then the priority of edge $\{u, v\}$ is

$$\eta_{\text{low}}(\{u, v\}) = -n \cdot \min(x_u, x_v) - \max(x_u, x_v)$$

In other words, $\{u, v\}$ is selected when u has the smallest x_u and v has the smallest x_v among the neighbours of u. In the left example of Figure 1, this strategy yields a matching of 6 nodes, which is maximum.

For the quality certification, we name C_M^+ the largest cover that we obtain among all bounding-heuristics, and we use it to lower-bound the minimum size $|C^*|$ of any vertex cover.

3.2.3 Solution-heuristics. To obtain a small vertex cover, it is possible to use matchings again. Finding a minimum maximal matching is NP-hard, but Algorithm 2 can provide a small matching with a high-degree first priority function η_{high}. With the definition of x_u as above, the priority of edge $\{u, v\}$ is

$$\eta_{\text{high}}(\{u, v\}) = n \cdot \max(x_u, x_v) + \min(x_u, x_v)$$

It means that $\{u, v\}$ is selected when u has the largest x_u and v has the largest x_v among neighbours of u. In the middle example of Figure 1, this yields a cover of 4 nodes, which is not optimum but significantly smaller than with the η_{low} function.

No constant approximation bound under 2 is known for vertex cover on general instances. Yet, there exist other heuristics without theoretical approximation guarantees that may result in much smaller covers in practice. A lot of attention has been given to the design of such heuristics, and the purpose of this paper is not to improve on existing solutions. A simple method, called node-greedy, is shown in Algorithm 3: it is similar to Algorithm 2, except that

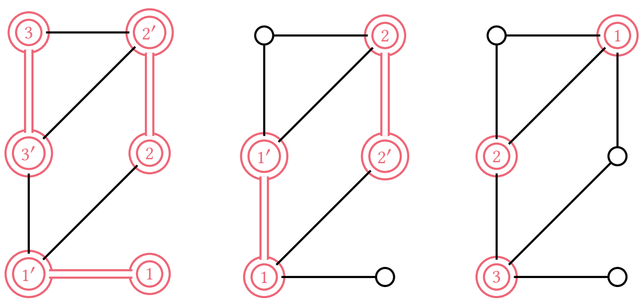

Figure 1: Greedy heuristics for vertex cover. The double lines show the cover obtained on a toy graph with three distinct heuristics; the numbers indicate the step at which a node has been selected. Left: edge-greedy with low-degree first η_{low} leads to a maximum matching. Middle: edge-greedy with high-degree first η_{high} gives a matching with 4 nodes. Right: node-greedy with high-degree first ν_{high} gives a cover of 3 nodes; it is optimum because the left cover has 6 nodes and is a 2-approximation.

it only adds node u at line 4. The priority function is now defined over the nodes, and a high-degree first priority is simply:

$$\nu_{\text{high}}(u) = x_u$$

In the right example of Figure 1, it yields a cover of 3 nodes, which we can certify is optimum, as there exists a 2-approximation with 6 nodes. Such greedy algorithms are known to be scalable and to give small covers [2, 21], and they become highly accurate when extra reduction rules and local search heuristics are added [12].

Algorithm 3 – Node-greedy heuristic for vertex cover

Input: priority function $\nu : V \to \mathbb{Z}$
1: start with empty cover: $C_H \leftarrow \emptyset$
2: **while** there are uncovered edges **do**
3: take node u with highest priority $\nu(u)$
4: add u to cover: $C_H \leftarrow C_H \cup \{u\}$
5: **return** C_H

For the quality certification, we name C_H^- the smallest cover that we obtain among all solution-heuristics, and we use it as an approximation of the minimum size $|C^*|$ of any vertex cover. In the end, we want to tighten the bounds of the resulting inequality:

$$\frac{1}{2}|C_M^+| \leq |C^*| \leq |C_H^-|$$

Definition 3.2 (Vertex cover certificate). Given a graph, a vertex cover certificate is a couple (C_H^-, C_M^+), where C_H^- is a (small) vertex cover of h nodes and C_M^+ is a vertex cover of $2b$ nodes that corresponds to a maximal matching of b edges. C_H^- is certified to be within a factor $\frac{h}{b}$ of the minimum cover size and the ratio $\frac{h}{b}$ is called the certified quality.

Note that this strategy can be interpreted as a primal-dual optimisation, as maximum matching is the dual linear program of vertex cover.

3.3 Certifying independent set

The case of independent sets is different because there is no known constant-factor approximation algorithm. It is thus necessary to use another method for the bounding. Besides, it is a maximisation problem, as we want to find as many non-adjacent nodes as possible; its approximation ratio will therefore be denoted by a factor between 0 and 1.

3.3.1 Solution-heuristics. Specific heuristics exist to obtain quick approximate solutions for independent set, in particular some that have been adapted from exact branch-and-bound algorithms [26].

Moreover, any solution-heuristic for vertex cover translates into a solution-heuristic for independent set, since they are complementary problems as noted in Section 2. More precisely, for any vertex cover C, its complement $S = V \setminus C$ is an independent set. For optimal values, the size s^* of a maximum independent set and the size c^* of a minimum vertex cover satisfy $s^* + c^* = n$. Take the smallest vertex cover obtained by solution-heuristics; if it has h nodes, then its complement is an independent set of $s = n - h$ nodes. Besides, the difference between h and c^* is equal to the difference between s and s^*.

3.3.2 Bounding-heuristics. Independent set is a maximisation problem so the quality certification requires an upper-bound. To obtain one, we transform a lower-bound of minimum vertex cover: suppose that a bounding-heuristic for vertex cover produces a bound $b \leq c^*$, then we have $n - b \geq n - c^* = s^*$. In the end, $n - b$ is an upper-bound for s^*.

Definition 3.3 (Independent set certificate). Given a graph, an independent set certificate is a couple (S^+, C_M^+), where S^+ is a (large) independent set of s nodes and C_M^+ is a vertex cover of $2b$ nodes that corresponds to a maximal matching of b edges. S^+ is certified to be within a factor $\frac{s}{n-b}$ of the maximum independent set size and the ratio $\frac{s}{n-b}$ is called the certified quality.

Note that the certified quality $\frac{s}{n-b}$ depends on the number of nodes n. It can thus be arbitrarily low, which is compatible with the fact that independent set does not have constant-factor polynomial approximations.

4 EXPERIMENTS

4.1 Experimental setup

Datasets. To measure the performance of the quality certification method, we apply it on a set of social networks (blogs, twitter, facebook, etc) and web graphs (wikipedia pages, webpages of a linguistic region, top-level domain, etc). For comparison purposes, we use all the undirected graphs of the Network Repository [29] anaysed in the experiments of [12], including other types of real-world networks (biological, citation, infrastructure). To test the limits of the different algorithms in the case of large online platforms, we add 10 networks of the *massive* category of [29], and two networks of the Webgraph [9]. All 114 networks are reported in Appendix A with their number of nodes and edges, the exact or approximate covers found by experiments and the corresponding quality certificates. We also report the duration of the certification to show its scalability.

Software and hardware. To emulate an online-application setting where users cannot tolerate a long latency, each execution is limited to one hour for the networks used in [12] and six hours to the 12 added networks. We release an open-source `c++` implementation of our code for the quality certification of vertex cover and independent set [2]. We run all the programs on a `sgi ub2000 intel xeon e5-4650L @2.6 GHz` of memory running `linux suse 12.3` with 128GB of memory.

4.2 Algorithms

4.2.1 Solution-heuristics for vertex cover and independent set. The literature presents specific methods and implementations for independent set [26], but most techniques have now been applied for vertex cover as well [18, 23]. Therefore, we only use the heuristics for vertex cover as their result directly translates into a result for independent set.

We try different methods to find a balance between cover quality and execution time. First, we allocate 1 hour (6 hours for larger networks) to the exact kernelisation algorithm presented in [23]; we obtain 76 *solved* networks where the optimum is known (several of which have millions of nodes), and 38 *unsolved* ones. For all the networks, we obtain fast approximate results with the state-of-the-art heuristic FastVC presented in [12]; on top of the one-hour time limit, we stop the execution when the best cover stops improving over m iterations.

We also use the node-greedy Algorithm 3 with high-degree first, as described on the right of Figure 1. To ensure that the resulting cover is minimal, we remove the nodes that have all of their neighbours in the cover. This algorithm is not a 2-approximation but it gives better results than the edge-greedy heuristic on all instances. It even yields smaller covers than FastVC on 18 networks. The experimental cover size taken as a reference in the experiments is the smallest between node-greedy and FastVC results.

4.2.2 For maximum matching. To find a lower-bound on the minimum size of a vertex cover, we need a poor cover approximation, which means a matching of high cardinality. An exact maximum matching can be found with the blossom algorithm. We use the implementation of the `boost`[3] `c++` library, and it finishes in less than an hour on all but 11 networks. For faster results, we use the edge-greedy Algorithm 2 as an approximation, with the low-degree priority function shown on the left of Figure 1. The resulting matching is within 2% of maximum on all the instances where the maximum is known.

Altogether, the quality certification method scales on large real-world datasets and can use more precise methods on smaller instances. Appendix A presents the execution times that lead to the results of the experiments.

4.3 Quality certification in practice

The main contribution of this work is to provide a scalable way to certify the quality of a vertex cover that is not known to be optimal. On each graph, we are therefore interested in the gap between the best cover obtained with a solution-heuristic (called C_H^- in Section 3)

[2]http://github.com/ (hidden for anonymity)

[3]https://boost.org/doc/libs/1_80_0/libs/graph/doc/maximum_matching.html

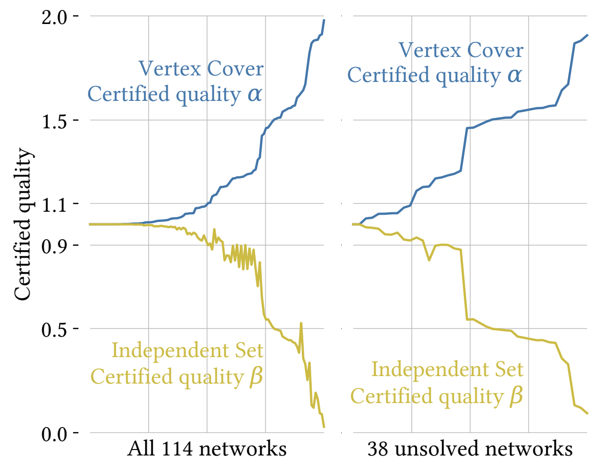

Figure 2: Certified quality for vertex cover and independent set. Left: all 114 networks. Right: 38 unsolved networks (exact solution unknown). Networks are ranked by their certified quality for vertex cover, which is correlated but not necessarily symmetric to the quality for independent set. Horizontal grids indicate 10% and 50% thresholds and vertical grids cut networks in four even groups. Observe that half of the networks obtain a certified quality within the 10% lines: the bounding-heuristic certifies that the smallest cover found by solution-heuristics is at most 10% larger than minimum, and that the corresponding independent set is at most 10% smaller than maximum.

and the highest bound obtained with a bounding-heuristic (called C_M^+). More specifically, the certified qualities α for vertex cover and β for independent set are given by

$$\alpha = \frac{|C_H^-|}{\frac{1}{2}|C_M^+|} \qquad\qquad \beta = \frac{n - |C_H^-|}{n - \frac{1}{2}|C_M^+|}$$

Recall that these two values are correlated, but while the vertex cover certificate is always between one and two, the certificate for independent set depends on n and is between zero and one.

We define the following arbitrary thresholds: a certified quality for vertex is good when below 1.1, poor when above 1.5, and medium otherwise. Similarly for independent set, it is good above 0.9, poor under 0.5, and medium otherwise.

Figure 2 shows that one half of the networks have a good certified quality for vertex cover: the solution-heuristics found a cover that is less than 10% larger than a minimum cover. The quality is even under 1.01 for 30 networks, and exact for 9 of them: a solution-heuristic found a cover C_H of h nodes, and a bounding-heuristic found a cover C_M of $2h$ nodes, certifying that h is the optimum value. Among unsolved networks, 14 have a good quality for independent set and 22 have a medium or good quality for vertex cover.

Altogether, this indicates that existing heuristics perform well, and that we are able to certify it using a maximal matching as a bounding-heuristics. However, 25 networks have a poor quality for both vertex cover and independent set. To understand this

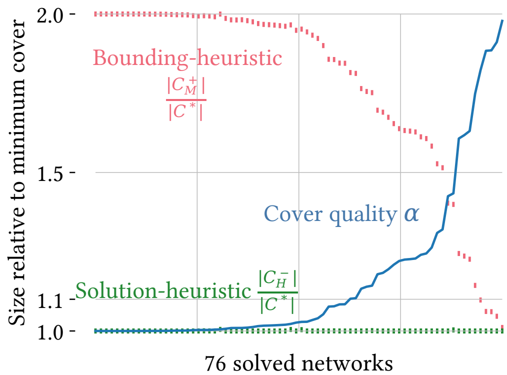

Figure 3: Size of the smallest cover found by solution-heuristics and of the largest cover found by bounding-heuristics, relative to known minimum cover size. Only solved networks are shown, and they are ranked by their certified quality. The cover of the solution-heuristic is always less than 1% above optimum, which shows that heuristics perform extremely well on all of these instances. The cover of the bounding-heuristic (obtained from a maximal matching) is almost twice as large as optimum in half of the cases. But on the other half, it is much smaller; this fully explains the poor approximation certificate for these networks.

unsatisfactory behaviour, Section 4.4 will check whether the certificate is poor because of the solution-heuristics or because of the bounding-heuristics.

Besides, we observe that the certificate β for independent set is precisely linked with the certificate for vertex cover α, almost following the relation $\alpha + \beta \simeq 2$. Yet, this equation is only valid when both β and α equal 1 (when an optimal cover has been found) or when the cover C_M of the maximum matching has n nodes. The correlation of the two curves indicates that we are close to one of these situations; this will be further investigated in Section 4.5.

4.4 Finding the cause for poor certified quality

As reported in Figure 2, the quality certification method gives poor results for 25 datasets, and this can have two causes. First, the solution-heuristics may find covers that are much larger than the optimum c^*. Second, the cover of a maximal matching (found by a bounding-heuristic) may be much smaller than $2c^*$. To measure this, we consider the 76 solved networks, for which an optimum cover has been found in less than an hour.

Looking at the solution-heuristic cover size in Figure 3, we observe that it is always very close to optimum regardless of the certified quality. In fact, the best heuristic finds a cover that is less than 1% larger than optimum on all solved networks. This means that the medium and poor qualities obtained on half of the networks are mainly due to a poor bound. We indeed see in the figure that bounding-heuristic size is almost twice the optimum size for half of the networks, but it drops for those of poor certified quality. On the last 10 networks, the cover of the matching found by bounding-heuristics is barely larger than the smallest cover found by solution-heuristics: this gives no indication at all on the quality of this cover.

The solution-heuristics that find small covers are not responsible for the poor certified qualities. It is rather the bounding-heuristics that fail because they find matchings that are too small. Three possibilities can explain this, and the experiment of Section 4.5 will find the culprit.

4.5 Finding the cause for poor bounds

Previous experiments show that bounding-heuristics are responsible for the medium or poor quality certifications obtained on half of the networks. Indeed, the cover of the matching that they find has much less than $2c^*$ nodes, which can have three causes.

First, the edge-greedy Algorithm 2 may output a cover that corresponds to a matching that is far from maximum; this is not the case, because the we were able to obtain an exact maximum matching on all the instances that have a poor certified quality (see the table in Appendix A). Second, the cover of the maximum matching may have much less than n nodes, which happens for example with isolated triangles, where matching two nodes leaves one alone. Third, the minimum cover itself contains almost n nodes, which makes it impossible to find a cover with $2n$ nodes.

Looking back at the relation $\alpha + \beta \simeq 2$ mentioned in Section 4.3 points to the third option. Rewriting it as $\frac{|C_H^-|}{\frac{1}{2}|C_M^+|} + \frac{n-|C_H^-|}{n-\frac{1}{2}|C_M^+|} = 2$, we have two solutions: either $|C_M^+| = 2|C_H^-|$, which certifies that C_H^- is a minimum vertex cover; or $|C_M^+| = n$, which is the third option to explain the poor qualities.

To attest this, let us take the 57 networks that have medium or poor certified cover qualities (above 1.1) and compare n to the largest cover obtained by bounding-heuristics. Figure 4 shows that in most cases, the largest cover obtained from a maximal matching contains almost n nodes, represented by the darkest area. Half of the matchings contain more of 90% of the nodes; only 12 are not in the 80%, and their bounding-heuristic size is more than 1.5 larger than the solution-heuristic, which means that their certified quality is medium.

Altogether, this result means that the quality certification only fails when the minimum cover of a graph contains almost all the nodes, because it loosens the bound obtained with a poor 2-approximation. Improving the certified quality on these networks would require a more advanced bounding strategy, using for instance the $\frac{3}{2}$-approximation presented in [3].

CONCLUSION

This work proposes a practical method to certify the quality of the result of a heuristic on a given instance. It then illustrates it on the closely related problems of vertex cover and independent set. For the former, our method certifies quality ratios that are way under the theoretical factor 2, even on networks with billions of edges where obtaining an exact solution is costly or unfeasible. For the latter, the resulting certificates are even more remarkable as the independent set problem does not have constant-factor approximations in general. However, the method obtains poor results on networks where the minimum cover contains most of the nodes,

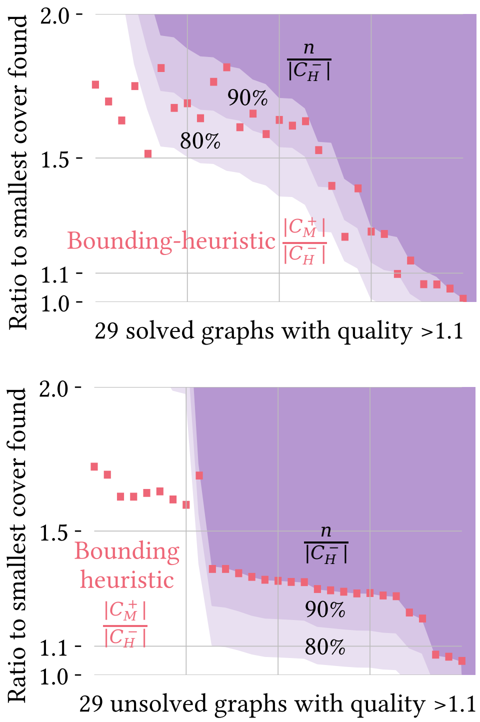

Figure 4: Size of the largest cover C_M^+ found by bounding-heuristics relative to smallest cover C_H^- found by solution-heuristic, compared to n. Solved (top) and unsolved (bottom) networks are shown if they have a medium or poor certified cover quality (above 1.1). Bounding-heuristic size and n are divided by the smallest size $|C_H^-|$. A large cover cannot contain more than n nodes, which is represented by the dark area; 90% and 80% zones show how far from n a given bounding-heuristic is. Observe that most of them contain more than 90% of the nodes, which means that the poor quality is due to $|C_H^-|$ being close to n. This is even more striking for unsolved networks, for 20 of which C_M^+ contains more than 99% of the nodes.

which invites for more research on the types of graph structures that influence the quality certification.

We hope that this work is an incentive to design certificates to go along with other heuristics. Further research will try and extend this method to other algorithmic problems. Indeed, the quality certification applies on all problems that have a constant-factor approximation – like vertex cover – or that can be expressed as a transformation of problems that have one – like independent set.

ACKNOWLEDGMENTS

Grateful thanks to Mathias Lécuyer, who first heard of this project and gave it the much-needed impetus. Sincere thanks to Lionel Tabourier for his careful review. Thanks as well to Alexis Baudin, Esteban Bautista, Mehdi Naima, Clémence Magnien and David Saulpic for their early feedback.

REFERENCES

[1] Eleni C Akrida. 2020. Temporal vertex cover with a sliding time window. *J. Comput. System Sci.* (2020), 16.

[2] Eric Angel, Romain Campigotto, and Christian Laforest. 2012. Implementation and Comparison of Heuristics for the Vertex Cover Problem on Huge Graphs. In *Experimental Algorithms*. Vol. 7276. Springer, 39–50. https://doi.org/10.1007/978-3-642-30850-5_5

[3] Eyjolfur Asgeirsson and Cliff Stein. 2007. Vertex Cover Approximations on Random Graphs. In *Experimental Algorithms*. Vol. 4525. Springer, 285–296. https://doi.org/10.1007/978-3-540-72845-0_22

[4] Sepehr Assadi, Krzysztof Onak, Baruch Schieber, and Shay Solomon. 2018. Fully dynamic maximal independent set with sublinear update time. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC 2018)*. ACM, New York, NY, USA, 815–826. https://doi.org/10.1145/3188745.3188922

[5] David Avis and Tomokazu Imamura. 2007. A list heuristic for vertex cover. *Operations Research Letters* 35, 2 (March 2007), 201–204. https://doi.org/10.1016/j.orl.2006.03.014

[6] Abbas Bazzi, Samuel Fiorini, Sebastian Pokutta, and Ola Svensson. 2018. No Small Linear Program Approximates Vertex Cover Within a Factor 2-epsilon. *Mathematics of OR* (Sept. 2018), moor.2017.0918. https://doi.org/10.1287/moor.2017.0918

[7] Norbert Blum. 1990. A new approach to maximum matching in general graphs. In *Automata, Languages and Programming (Lecture Notes in Computer Science)*, Michael S. Paterson (Ed.). Springer, Berlin, Heidelberg, 586–597. https://doi.org/10.1007/BFb0032060

[8] Thomas Bläsius, Philipp Fischbeck, Tobias Friedrich, and Maximilian Katzmann. 2020. *Solving Vertex Cover in Polynomial Time on Hyperbolic Random Graphs*. Technical Report arXiv:1904.12503. arXiv. http://arxiv.org/abs/1904.12503

[9] Paolo Boldi and Sebastiano Vigna. 2004. The webgraph framework I: compression techniques. In *WWW*. 1. https://law.di.unimi.it/datasets.php https://law.di.unimi.it/datasets.php.

[10] Endre Boros and Vladimir Gurvich. 2007. On complexity of algorithms for modeling disease transmission and optimal vaccination strategy. (2007), 12 pages.

[11] Shaowei Cai, Wenying Hou, Jinkun Lin, and Yuanjie Li. 2018. Improving Local Search for Minimum Weight Vertex Cover by Dynamic Strategies. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, Stockholm, Sweden, 1412–1418. https://doi.org/10.24963/ijcai.2018/196

[12] Shaowei Cai, Jinkun Lin, and Chuan Luo. 2017. Finding A Small Vertex Cover in Massive Sparse Graphs: Construct, Local Search, and Preprocess. *Journal of Artificial Intelligence Research* 59 (July 2017), 463–494. https://doi.org/10.1613/jair.5443 Code provided at http://lcs.ios.ac.cn/~caisw/VC.html.

[13] Irit Dinur and Samuel Safra. 2005. On the Hardness of Approximating Minimum Vertex Cover. *Annals of Mathematics* 162, 1 (2005), 439–485. https://www.jstor.org/stable/3597377

[14] Ran Duan and Seth Pettie. 2014. Linear-Time Approximation for Maximum Weight Matching. *J. ACM* 61, 1 (Jan. 2014), 1–23. https://doi.org/10.1145/2529989

[15] M. Ayaz Dzulfikar, Johannes K. Fichte, and Markus Hecher. 2019. *The PACE 2019 Parameterized Algorithms and Computational Experiments Challenge: The Fourth Iteration (Invited Paper)*. Technical Report. 23 pages pages. http://drops.dagstuhl.de/opus/volltexte/2019/11486/

[16] Jack Edmonds. 1965. Paths, Trees, and Flowers. *Canadian Journal of Mathematics* 17 (1965), 449–467. https://doi.org/10.4153/CJM-1965-045-4

[17] David Eppstein, Maarten Löffler, and Darren Strash. 2013. Listing All Maximal Cliques in Large Sparse Real-World Graphs. *ACM Journal of Experimental Algorithms* 18 (Dec. 2013). https://doi.org/10.1145/2543629

[18] Michael R. Fellows, Lars Jaffke, Aliz Izabella Király, Frances A. Rosamond, and Mathias Weller. 2018. What Is Known About Vertex Cover Kernelization? In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. Vol. 8827. Springer, 330–356. https://doi.org/10.1007/978-3-319-98355-4_19

[19] Mikael Gast and Mathias Hauptmann. 2014. Approximability of the vertex cover problem in power-law graphs. *Theoretical Computer Science* 516 (Jan. 2014), 60–70. https://doi.org/10.1016/j.tcs.2013.11.004

[20] Mohsen Ghaffari, Themis Gouleakis, Christian Konrad, Slobodan Mitrović, and Ronitt Rubinfeld. 2018. Improved Massively Parallel Computation Algorithms for MIS, Matching, and Vertex Cover. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing (PODC '18)*. ACM, New York, NY, USA, 129–138. https://doi.org/10.1145/3212734.3212743

[21] Fernando C. Gomes, Cláudio N. Meneses, Panos M. Pardalos, and Gerardo Valdisio R. Viana. 2006. Experimental Analysis of Approximation Algorithms for the Vertex Cover and Set Covering Problems. *Computers & Operations Research* 33, 12 (Dec. 2006), 3520–3534. https://doi.org/10.1016/j.cor.2005.03.030

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

[23] Demian Hespe, Sebastian Lamm, Christian Schulz, and Darren Strash. 2020. WeGotYouCovered: The Winning Solver from the PACE 2019 Challenge, Vertex Cover Track. In *Workshop on Combinatorial Scientific Computing (CSC)*. SIAM, 1–11. https://doi.org/10.1137/1.9781611976229.1 Code available at https://zenodo.org/record/2816116.

[24] Dorit S. Hochbaum. 1998. The t-vertex cover problem: Extending the half integrality framework with budget constraints. In *Approximation Algorithms for Combinatiorial Optimization (Lecture Notes in Computer Science)*, Klaus Jansen and José Rolim (Eds.). Springer, Berlin, Heidelberg, 111–122. https://doi.org/10.1007/BFb0053968

[25] Subhash Khot and Oded Regev. 2008. Vertex cover might be hard to approximate to within 2-epsilon. *J. Comput. System Sci.* 74, 3 (May 2008), 335–349. https://doi.org/10.1016/j.jcss.2007.06.019

[26] Sebastian Lamm, Peter Sanders, Christian Schulz, Darren Strash, and Renato F. Werneck. 2017. Finding near-optimal independent sets at scale. *J Heuristics* (2017). https://doi.org/10.1007/s10732-017-9337-x Code provided at https://karlsruhemis.github.io.

[27] Silvio Micali and Vijay V. Vazirani. 1980. An O(v|v| c |E|) algoithm for finding maximum matching in general graphs. In *21st Annual Symposium on Foundations of Computer Science*. 17–27. https://doi.org/10.1109/SFCS.1980.12

[28] Pooja Pandey and Abraham P. Punnen. 2018. The generalized vertex cover problem and some variations. *Discrete Optimization* 30 (Nov. 2018), 121–143. https://doi.org/10.1016/j.disopt.2018.06.004

[29] Ryan A. Rossi and Nesreen K. Ahmed. 2015. The Network Data Repository with Interactive Graph Analytics and Visualization. In *AAAI*. https://networkrepository.com.

[30] Pedro V. Sander, Diego Nehab, Eden Chlamtac, and Hugues Hoppe. 2008. Efficient traversal of mesh edges using adjacency primitives. *ACM Trans. Graph.* 27, 5 (Dec. 2008), 1–9. https://doi.org/10.1145/1409060.1409097

[31] E. Sáenz-de Cabezón and H.P. Wynn. 2014. Measuring the robustness of a network using minimal vertex covers. *Mathematics and Computers in Simulation* 104 (Oct. 2014), 82–94. https://doi.org/10.1016/j.matcom.2014.04.001

[32] Mingyu Xiao, Sen Huang, Yi Zhou, and Bolin Ding. 2021. Efficient Reductions and a Fast Algorithm of Maximum Weighted Independent Set. In *Proceedings of the Web Conference 2021 (WWW '21)*. ACM, New York, NY, USA, 3930–3940. https://doi.org/10.1145/3442381.3450130

[33] Yasin Yigit, Vahid Khalilpour Akram, and Orhan Dagdeviren. 2021. Breadth-first search tree integrated vertex cover algorithms for link monitoring and routing in wireless sensor networks. *Computer Networks* 194 (July 2021), 108144. https://doi.org/10.1016/j.comnet.2021.108144

A DATASETS AND EXHAUSTIVE RESULTS

Tables 1 (social and web networks) and 2 (other networks) present all 114 real-world graphs with their names, number of nodes and edges. The choice of networks came from [12], but we added large networks of the *massive* category of [29] and two networks of [9]. The next columns display: the minimum vertex cover size if found in one hour († otherwise); the smallest cover size found by solution-heuristics; the largest cover found by bounding-heuristics through blossom algorithm (or through edge-greedy Algorithm 2) when there is a †). The certified quality for vertex cover (VC) and independent set (IS) is better when closer to 1, and a 1 proves that an optimum solution has been found to both problems. The last column shows the time in seconds of the certification quality: it is the sum of the best solution-heuristic and the best bounding-heuristic that finished on this network.

Table 1: Results of the experiments on web and social networks

			minimum	VC heuristic results		certified quality		time
name	**nodes**	**edges**	**vertex cover**	**solution**	**bounding**	**VC**	**IS**	**(seconds)**
rt-retweet	97	117	32	32	64	1	1	<0.1
rt-twitter-copen	762	1,029	237	237	466	1.017	0.992	<0.1
rt-retweet-crawl	1,112,703	2,278,852	81,040	81,048	162,074	<1.0005	>0.9995	19.1
soc-karate	35	78	14	14	26	1.077	0.955	<0.1
soc-dolphins	63	159	34	34	60	1.133	0.879	<0.1
soc-wiki-Vote	890	2,914	406	406	802	1.012	0.990	<0.1
soc-epinions	26,589	100,120	9,757	9,757	19,136	1.020	0.989	0.58
soc-brightkite	56,740	212,945	21,190	21,190	41,600	1.019	0.989	1.62
soc-douban	154,909	327,162	8,685	8,685	17,370	1	1	1.24
soc-slashdot	70,069	358,647	22,373	22,373	44,332	1.009	0.996	2.17
soc-twitter-follows	404,720	713,319	2,323	2,323	4,646	1	1	3.04
soc-gowalla	196,592	950,327	84,222	84,226	162,808	1.035	0.976	28.4
soc-delicious	536,109	1,365,961	85,298	85,717	170,572	1.005	0.999	22.6
soc-youtube	495,958	1,936,748	146,376	146,376	289,674	1.011	0.996	99.0
soc-BlogCatalog	88,785	2,093,195	20,752	20,752	41,322	1.004	0.999	13.5
soc-LiveMocha	104,104	2,193,083	43,427	43,427	86,754	1.001	0.999	34.5
soc-buzznet	101,164	2,763,066	30,613	30,626	60,670	1.010	0.996	33.9
soc-youtube-snap	1,134,891	2,987,624	276,945	276,945	548,662	1.010	0.997	196
soc-flickr	513,970	3,190,452	153,271	153,272	298,654	1.026	0.989	95.9
soc-FourSquare	639,015	3,214,986	90,108	90,109	179,644	1.003	0.999	54.9
soc-lastfm	1,191,806	4,519,330	78,688	78,688	157,336	<1.0005	>0.9995	32.8
soc-digg	770,800	5,907,132	103,234	103,245	205,820	1.003	0.999	160
soc-flixster	2,523,387	7,918,801	96,317	96,317	192,596	<1.0005	>0.9995	45.7
soc-pokec	1,632,804	22,301,964	†	843,440	1,562,088	1.080	0.927	3306
soc-livejournal	4,033,138	27,933,062	1,868,903	1,869,045	3,551,580 †	1.053	0.959	1220
soc-orkut	2,997,167	106,349,209	†	2,171,318	2,971,206	1.462	0.546	2808
socfb-CMU	6,622	249,959	†	4,989	6,602	1.511	0.492	1.08
socfb-MIT	6,403	251,230	†	4,658	6,374	1.462	0.543	0.86
socfb-UCSB37	14,918	482,215	†	11,266	14,900	1.512	0.489	2.32
socfb-Duke14	9,886	506,437	†	7,686	9,864	1.558	0.444	1.83
socfb-Stanford3	11,587	568,309	†	8,518	11,530	1.478	0.527	2.09
socfb-UConn	17,207	604,867	†	13,234	17,186	1.540	0.461	2.22
socfb-UCLA	20,454	747,604	†	15,229	20,420	1.492	0.510	3.67
socfb-OR	63,393	816,886	†	36,556	61,886	1.181	0.827	5.90
socfb-Wisconsin87	23,832	835,946	†	18,398	23,810	1.545	0.456	3.34
socfb-Berkeley13	22,901	852,419	†	17,219	22,852	1.507	0.495	4.38
socfb-UIllinois	30,796	1,264,421	†	24,103	30,760	1.567	0.434	6.95
socfb-Indiana	29,733	1,305,757	†	23,323	29,706	1.570	0.431	6.99
socfb-Penn94	41,537	1,362,220	†	31,176	41,490	1.503	0.498	13.2
socfb-UF	35,112	1,465,654	†	27,316	35,092	1.557	0.444	13.9
socfb-Texas84	36,365	1,590,651	†	28,186	36,340	1.551	0.450	11.2
socfb-B-anon	2,937,613	20,959,854	303,048	303,049	605,978	<1.0005	>0.9995	918
socfb-A-anon	3,097,166	23,667,394	375,230	375,233	750,174	<1.0005	>0.9995	2721
socfb-uci-uni	58,790,783	92,208,195	866,766	866,768	1,733,530	<1.0005	>0.9995	1125
web-polblogs	644	2,280	244	244	480	1.017	0.990	<0.1
web-google	1,300	2,773	498	498	812	1.227	0.897	<0.1
web-edu	3,032	6,474	1,451	1,451	2,820	1.029	0.975	<0.1
web-BerkStan	12,306	19,500	5,384	5,390	9,432	1.143	0.911	0.19
web-webbase-2001	16,063	25,593	2,651	2,652	4,654	1.140	0.976	0.11
web-spam	4,768	37,375	2,297	2,298	4,264	1.078	0.937	0.13
web-indochina-2004	11,359	47,606	7,300	7,300	10,244	1.425	0.651	0.27
web-sk-2005	121,423	334,419	58,173	58,181	88,130	1.320	0.818	4.45
web-arabic-2005	163,599	1,747,269	114,420	114,430	140,326	1.631	0.526	17.9
web-wikipedia2009	1,864,434	4,507,315	†	648,333	1,261,130	1.028	0.986	3006
web-it-2004	509,339	7,178,413	414,507	414,676	455,018	1.823	0.336	59.8
web-uk-2005	129,633	11,744,049	127,774	127,774	129,180	1.978	0.029	74.2
web-indochina-2004-all	7,414,866	150,984,819	†	2,720,341	4,405,674 †	1.235	0.901	1305
+web-indochina-2004-all	7,414,866	150,984,819	†	2,720,245	4,405,674 †	1.235	0.901	3756
+soc-sinaweibo	58,655,850	261,321,033	†	223,000	446,000	1	1	3963
+web-uk-2002-all	18,520,344	261,787,258	†	6,630,435	11,244,516 †	1.179	0.922	1429
+soc-twitter-2010	21,297,773	265,025,545	†	7,646,447	14,825,238 †	1.032	0.983	1545
+web-uk-2005-all	39,459,924	783,027,125	†	15,952,499	25,384,548 †	1.257	0.878	327
+web-webbase-2001-all	118,142,144	854,809,761	†	38,892,190	67,042,152 †	1.160	0.937	503
+web-it-2004-all	41,291,319	1,027,474,947	†	15,986,216	26,183,814 †	1.221	0.897	392
+soc-friendster	65,608,367	1,806,067,135	†	29,609,968	56,158,240 †	1.055	0.959	2456
+web-sk-2005-all	50,636,152	1,810,063,330	†	20,352,253	32,764,424 †	1.242	0.884	661
+webgraph-twitter-2010	41,652,230	1,202,513,046	†	13,066,103	24,884,176 †	1.050	0.979	1326
+webgraph-uk-2007-05	105,896,435	3,301,876,564	†	39,386,701	64,299,206 †	1.225	0.902	1663

Table 2: Results of the experiments on other real-world networks

			minimum	VC heuristic results		certified quality		time
name	**nodes**	**edges**	**vertex cover**	**solution**	**bounding**	**VC**	**IS**	**(seconds)**
bio-diseasome	517	1,188	285	285	458	1.245	0.806	<0.1
bio-yeast	1,459	1,948	456	456	896	1.018	0.992	<0.1
bio-celegans	454	2,025	249	249	452	1.102	0.899	<0.1
bio-dmela	7,394	25,569	2,630	2,630	5,260	1	1	<0.1
ca-netscience	380	914	214	214	354	1.209	0.818	<0.1
ca-CSphd	1,883	1,740	550	550	1,100	1	1	<0.1
ca-Erdos992	6,101	7,515	461	461	922	1	1	<0.1
ca-GrQc	4,159	13,422	2,208	2,208	3,732	1.183	0.851	<0.1
ca-CondMat	21,364	91,286	12,480	12,480	20,372	1.225	0.795	0.50
ca-HepPh	11,205	117,619	6,555	6,555	10,570	1.240	0.785	0.54
ca-AstroPh	17,904	196,972	11,483	11,483	17,540	1.309	0.703	1.00
ca-dblp-2010	226,414	716,460	121,969	121,969	199,818	1.221	0.826	19.7
ca-citeseer	227,321	814,134	129,193	129,193	204,536	1.263	0.785	19.8
ca-MathSciNet	332,690	820,644	139,951	139,951	258,144	1.084	0.947	51.0
ca-dblp-2012	317,081	1,049,866	164,949	164,949	276,148	1.195	0.850	57.5
ca-coauthors-dblp	540,487	15,245,729	472,179	472,179	540,080	1.749	0.253	134
ca-hollywood-2009	1,069,127	56,306,653	864,052	864,052	1,068,158	1.618	0.383	792
ia-enron-only	144	623	86	86	140	1.229	0.784	<0.1
ia-infect-hyper	114	2,196	90	90	112	1.607	0.414	<0.1
ia-infect-dublin	411	2,765	293	294	410	1.434	0.568	<0.1
ia-email-univ	1,134	5,451	594	594	1,096	1.084	0.922	<0.1
ia-fb-messages	1,267	6,451	578	578	1,148	1.007	0.994	<0.1
ia-reality	6,810	7,680	81	81	162	1	1	<0.1
ia-email-EU	32,431	54,397	820	820	1,638	1.001	>0.9995	0.17
ia-enron-large	33,697	180,811	12,781	12,781	21,682	1.179	0.915	1.04
ia-wiki-Talk	92,118	360,767	17,288	17,288	34,526	1.001	>0.9995	1.95
inf-power	6,595	6,548	2,186	2,187	4,360	1.003	0.998	<0.1
inf-roadNet-PA	1,087,563	1,541,514	†	555,276	1,057,094	1.051	0.952	1171
inf-roadNet-CA	1,957,028	2,760,388	†	1,001,358	1,902,836	1.052	0.950	3763
inf-road-usa	23,947,348	28,854,312	†	11,972,844	22,255,674 †	1.076	0.934	1255
rec-amazon	91,814	125,704	47,605	47,606	86,290	1.103	0.908	4.19
sc-nasasrb	54,871	1,311,227	†	51,251	54,870	1.868	0.132	19.3
sc-shipsec1	140,386	1,707,759	†	117,357	140,382	1.672	0.328	41.1
sc-shipsec5	179,105	2,200,076	†	147,161	179,094	1.643	0.357	52.0
sc-pkustk11	87,805	2,565,054	83,911	83,911	87,804	1.911	0.089	15.9
sc-pkustk13	94,894	3,260,967	†	89,228	94,892	1.881	0.119	33.0
sc-pwtk	217,892	5,653,221	†	207,721	217,890	1.907	0.093	89.7
sc-msdoor	415,864	9,378,650	381,558	381,558	404,784	1.885	0.161	122
sc-ldoor	952,204	20,770,807	856,754	856,758	909,536	1.884	0.192	439
tech-routers-rf	2,114	6,632	795	795	1,566	1.015	0.991	<0.1
tech-as-caida2007	26,476	53,381	3,683	3,683	7,360	1.001	>0.9995	0.20
tech-WHOIS	7,477	56,943	2,284	2,284	4,392	1.040	0.983	0.20
tech-internet-as	40,165	85,123	5,700	5,700	11,370	1.003	>0.9995	0.37
tech-p2p-gnutella	62,562	147,878	15,682	15,682	31,364	1	1	0.65
tech-RL-caida	190,915	607,610	74,593	74,942	146,626	1.022	0.986	24.5
tech-as-skitter	1,694,617	11,094,209	525,022	527,198	1,024,996	1.029	0.988	3154
+tech-p2p	5,792,297	147,829,887	†	301,718	603,432	<1.0005	>0.9995	5091