Journal of Algorithms **39**, 223–240 (2001)
doi:10.1006/jagm.2000.1155, available online at http://www.idealibrary.com on IDEAL®

Greedy Local Improvement and Weighted Set Packing Approximation[1]

Barun Chandra

Department of Computer Science, University of New Haven, West Haven, Connecticut 06516
E-mail: barun@charger.newhaven.edu

and

Magnús M. Halldórsson

Science Institute, University of Iceland, IS-107 Reykjavik, Iceland
E-mail: mmh@hi.is

Received June 27, 2000

Given a collection of weighted sets, each containing at most k elements drawn from a finite base set, the k-set packing problem is to find a maximum weight subcollection of disjoint sets. A greedy algorithm for this problem approximates it to within a factor of k, and a natural local search has been shown to approximate it to within a factor of roughly $k - 1$. However, neither paradigm can yield approximations that improve on this.

We present an approximation algorithm for the weighted k-set packing problem that combines the two paradigms by starting with an initial greedy solution and then repeatedly choosing the best possible local improvement. The algorithm has a performance ratio of $2(k + 1)/3$, which we show is asymptotically tight. This is the first asymptotic improvement over the straightforward ratio of k.

Key Words: set packing, weighted independent set, local search, greedy algorithms, approximation algorithms.

1. INTRODUCTION

We consider the following problem:

Weighted k-Set Packing. Given a collection of sets, each of which has an associated real weight and contains at most k elements drawn from

[1] An extended abstract of this paper appeared at SODA '99.

0196-6774/01 $35.00

a finite base set, find a collection of disjoint sets of maximum total weight.

Set packing is a fundamental combinatorial problem that underlies a range of practical and theoretical problems. The restriction to sets of size at most k properly includes multidimensional matching, which is a generalization of the ordinary graph matching problem.

k-Set packing can be further generalized to a certain independent set problem. The intersection graph of a set system of k-sets has the property that no induced subgraph contains a $k+1$-*claw*, which is a set of $k+1$ independent (i.e. mutually nonadjacent) vertices that share a common neighbor. A set packing corresponds to an independent set in the intersection graph. The class of $(k+1)$-*claw*-free graphs, however, properly includes these intersection graphs.

These problems are $\mathcal{NP}$-hard for any $k \geq 3$, even in the unweighted case [6], thus we seek heuristics with guaranteed solution quality. The most natural heuristic for the Weighted k-Set Packing Problem is the *greedy* algorithm: Add to the current solution a set of maximum weight, remove it and all sets that intersect it, and repeat until all sets have been removed. It is easy to see that this solution is within a factor of k from optimal: from the sets removed in each iteration, the optimal solution can contain at most k sets (at most one for each element of our chosen set), all of which are of weight at most that of our chosen set. It is also easy to construct examples that show that this factor of k cannot be improved.

Another natural strategy is local search: Attempt to replace a small subset of the solution with some collection of greater total weight that does not intersect the remainder of the solution. For such a search to be polynomially bounded, it has to be restricted in some way, such as that either the number of sets added or the number of sets removed should be constant. In the case of unweighted sets, Hurkens and Schrijver [8] showed that a local search algorithm leads to an approximation of $k/2+\epsilon$, where $\epsilon > 0$ depends on the size of the change set. This is the best performance ratio known to date for the unweighted k-set packing problem. A restricted form of this local search was considered in [7] with the same performance but decreased complexity.

Local search for the weighted case was independently analyzed by Arkin and Hassin [1] and by Bafna *et al*. [2]. They showed that a local search algorithm (which bounds the number of sets added) yields an approximation of $k-1+\epsilon$ and showed this to be a tight bound. Bafna *et al*. showed this for $\epsilon = 1/k$, while Arkin and Hassin showed this for any fixed ϵ.

The results proven for local search apply to a very general situation that contains much nondeterminism: any locally optimal solution achieves the given bound, independent of the starting solution or the particular sequence of feasible improvements. It leaves open the question of whether there exist easily computable rules for choosing a starting solution and for deciding among candidate improvements such that the resulting locally optimal solutions are guaranteed to be better. In particular, can one get an improved performance ratio by starting with a greedy solution and choosing improvements that yield bigger gains to the solution? Note that the latter is necessary, since it is possible to show that, even if starting with a greedy solution, indiscriminate choices of a improvements will lead to a solution no better than one for a pure local search.

We answer this question in the positive and obtain the first asymptotic improvement in the approximability of the problem. We present a natural heuristic BESTIMP that combines the greedy and local search paradigms by starting with an initial greedy solution and then repeatedly choosing the best possible local improvement. Its performance ratio is at most $2(k+1)/3$, which is asymptotically tight. This yields an improvement over a plain local search for $k \geq 5$.

In order to further examine the effect of the choice of the improvement (when more than one local improvement applies) on the quality of the solution, we consider another algorithm, ANYIMP, that also combines the greedy and local search paradigms. The difference is that ANYIMP just looks for an improvement that leads to a gain bigger than a specified threshold, instead of looking for the best improvement. The proof technique we use to obtain upper bounds on the performance ratio can be understood in a simpler setting with ANYIMP. We illustrate it by deriving an asymptotically tight bound on the performance ratio of ANYIMP as a function of the threshold, which for the best choice of a threshold is at most $(4k+2)/5$. It is interesting to find that picking the best improvement instead of a good enough improvement leads to a substantially better performance ratio. The results hold equally for the slightly more general problem of approximating maximum weight independent sets in $k+1$-claw free graphs.

Following the original appearance of a version of this paper [4], Berman [3] has closed the remaining gap, obtaining a $(k+1)/2$-approximation algorithm. His method is also based on local improvement, but one that is based on a sum of the *squares* of the weights of the vertices involved instead of the simple sum of weights considered here and before. This implies that even though greediness improves the performance of a basic local improvement as we have shown, it does not suffice to obtain the optimal performance ratio.

The organization of the rest of the paper is as follows. Section 2 contains formal definitions of the problem, statements of the algorithms, and bounds on their time complexity. The key concepts of the analysis are considered in Section 3 along with some of their properties. Section 3 also contains the analysis of the upper bound of ANYIMP, while the analysis of the upper bound of BESTIMP is presented in Section 4. A construction is given in Section 5 that establishes a lower bound on the performance ratio of BESTIMP. Section 6 ends with conclusions and open issues.

2. DEFINITIONS, NOTATION, AND PROBLEM STATEMENTS

The *weighted independent set* problem is as follows: Given a vertex-weighted graph $G = (V, E)$, find a maximum weight subset of mutually nonadjacent vertices, i.e., a subset $V' \subseteq V$ such that $v_i, v_j \in V'$ implies $(v_i, v_j) \notin E$.

The intersection graph of a set system contains a vertex for each set with edges between vertices whose corresponding sets intersect. If the sets are of size at most k, the intersection graph has the property that it contains no $k+1$-*claw*, i.e., a $k+1$-independent set in the neighborhood of some vertex. A *set packing* of a set system is a collection of mutually disjoint sets, and it corresponds to an independent set of the associated intersection graph. In the weighted version, sets in the set system have weights, which translate to vertex weights in the intersection graph. Since the independent set problem in claw-free graphs generalizes the set packing problem, we state our algorithms and results in terms of the former.

The approximation ratio $\rho(G)$ of a heuristic algorithm on a given graph G is the ratio between the size of the optimal solution and the size of the algorithm's solution on G. The *performance ratio* ρ of the algorithm is the maximum approximation ratio over all instances.

Let $G = (V, E)$ be a graph, $U, W \subseteq V$, $v \in V$. The neighborhood of v, $N(v)$, is the set of all vertices from V which are adjacent to v, and $deg(v) = |N(v)|$ is the degree of v. Define $N_W(v) = N(v) \cap W$ and $deg_W(v) = |N_W(v)|$. These definitions are extended to neighborhoods of sets of vertices: $N(U) = \bigcup_{u \in U} N(u)$, $N_W(U) = N(U) \cap W$. Note that if G is $k+1$-claw free and W is an independent set, then $deg_W(v) \le k$.

The Algorithms ANYIMP$_\alpha$ *and* BESTIMP

GREEDY is the natural greedy algorithm for the independent set problem which works by repeatedly picking the heaviest vertex from among the remaining vertices and eliminating it and the adjacent vertices. Let Gr denote the independent set selected by GREEDY on input graph G.

Our algorithms are based on the following type of a local improvement. An independent set I' is an *improvement* of I if, for some $x \in I$ and some independent set $Q \subseteq N(x)$, $I' = (I \cup Q) - N_I(Q)$. Namely, I' is formed by adding Q to I and removing those vertices of I which are adjacent to a vertex in Q. We define the *payoff factor* of the improvement to be

$$\frac{w(Q)}{w(N_I(Q))} = \frac{w(I' - I)}{w(I - I')}.$$

For $\alpha > 1$, an *α-improvement* is an improvement with payoff factor α, and an *α-good* improvement is an improvement with payoff factor at least α. A solution is α-locally optimal if it has no β-good improvement, for any $\beta > \alpha$, and is *locally optimal* if it is 1-locally optimal.

Both of our algorithms, AnyImp$_\alpha$ and BestImp, start with the initial greedy independent set Gr and repeatedly make local improvements until they reach a local optimum. In each iteration they find an improvement I' to the current solution I and then arbitrarily extend it to a maximal solution I''. The difference lies in which improvement is made: BestImp makes the improvement with the highest payoff factor, while AnyImp$_\alpha$ makes any α-good improvement.

Let us view the addition of vertices (to a nonmaximal independent set) as improvements of a nearly infinite payoff factor, further ordered according to the weight of the vertex added. Then, BestImp can be more succinctly stated as

“BESTIMP(G)
 $I \leftarrow \emptyset$
 while I is not locally optimal do
  Let I' be improvement of maximum payoff factor
  $I \leftarrow I'$
 od
 output I”

Complexity Analysis

The algorithms as presented are not polynomially bounded, since the improvements made can be arbitrarily small and the length of the improvement sequence can be exponential in the length of the input. In fact, the theory of $\mathcal{PLS}$-completeness indicates that finding locally optimal solutions to many weighted problems may not be much easier than finding optimal solutions [9]. We are, however, not interested in locally optimal solutions *per se*; rather, they are our vehicle for finding solutions with good performance bounds.

One solution to this problem is to truncate the weight of the vertices to integer multiples of $(1/n^2)w(Gr)$. Since we know that Gr is within a factor k from optimal, this limits the number of improvements to n^2k. This may reduce the weights of the optimal solution, and thus the guaranteed weight of the obtained solution, but only by a factor smaller than $n/(n-1)$.

The cost of each improvement step depends on the time it takes to explore all independent sets within the neighborhood of each vertex in the solution. In a given neighborhood, there are $\binom{\Delta}{i}$ subsets of size i or at most Δ^k sets of size at most k, where Δ is the maximum degree in the graph. Checking each for independence takes k^2 time for a combined complexity of $O(nk^2\Delta^k)$.

3. PROOF TECHNIQUE

In this section we describe the proof technique which we use to prove upper bounds on the performance ratios of AnyImp and BestImp and apply it to AnyImp. A central issue in the analysis of an approximation algorithm is to identify the computational structure that provides a bound on the optimal solution and its relationship with the heuristic solution. For example, Christofides' TSP heuristic uses both a minimum spanning tree and a minimum weight matching. In some cases, as in Christofides' heuristic, the structure is a part of the heuristic solution. In other cases, it is an entirely different beast, which may not even be polynomial computable, but in terms of which both the optimal and the heuristic solutions can be bounded. The projection plays this role in our analysis of AnyImp and BestImp.

The Projection. We introduce the concept of a *projection* of an optimal solution OPT to a given maximal solution I. Each vertex v of the graph is assigned a *representative*, which is the maximum weight vertex in I that is adjacent to v (or identical to v, when v is also in I). We are interested only in the representatives of vertices in OPT. The projection of OPT onto I is the multiset of the representatives of the vertices of OPT. We will overload the term to denote also its weight, i.e., the sum of the representatives' weights.

An example is given in Fig. 1, where I is a solution containing elements $a_1, a_2, \ldots, a_5$ ordered by nonincreasing weights. Bold lines indicate a representative relation, while dotted lines are other edges in the graph induced by $I \cup OPT$.

This projection is crucial to the analysis, acting as a stored potential or the unused capability of the algorithm's solution. We will show that the projection has certain properties, intuitively described as follows:

Property 1. The projection is initially as large as the optimal solution value.

Property 2. If the value of the projection goes down, the weight of the algorithm's current independent set goes up. The extent of this depends on the goodness of the improvement(s) made.

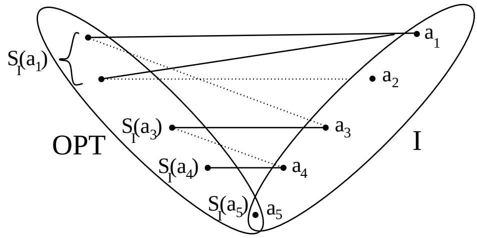

FIG. 1. Example of a projection of *OPT* onto the heuristic solution I.

Property 3. The weight of a locally optimal solution (i.e., the output of the algorithm) is large in comparison with the final projection.

We will use these properties to argue that the weight of the algorithm's solution is large compared to the optimal solution. The intuition is that if the projection does not decrease much from beginning to end, then by Property 3 the algorithm's solution is large compared to the projection which, by Property 1, is large compared to the optimal solution. On the other hand, if the value of the projection decreases a lot as the algorithm progresses, then by Property 2 the algorithm's solution improves a lot. In other words, either the initial greedy solution was already a good one or the final solution is a significant improvement on the greedy solution.

FORMAL DEFINITION. Let *OPT* be a particular independent set of maximum weight. We overload *OPT* to refer also to the weight of the set. We associate with each maximal independent set I a function $f_I : OPT \to I$. The representative of b, $f_I(b)$, is the vertex of maximum weight in $N_I(b)$ or b itself when b is in I. For an element v in I, the *preimage* of v is the set of elements that map to v, given by

$$S_I(v) = \{b \in OPT : f_I(b) = v\}.$$

The subscript is omitted when clear from context.

For an independent set I, the *projection* of *OPT* to I is defined as

$$proj(I) = \sum_{b \in OPT} w(f_I(b)) = \sum_{v \in I} |S_I(v)| w(v).$$

Properties of the Projection. Consider the greedy solution Gr. Note that the weight of any element of *OPT* is at most the weight of its representative, as otherwise the Greedy Algorithm would have chosen it instead of the representative. It follows that

$$proj(Gr) \geq OPT, \tag{1}$$

establishing Property 1 of the projection.

We can use this to derive easily a well-known bound on the greedy solution. Since the graph is $k+1$-claw free, any preimage contains at most k elements. Thus we have, for any I, that $proj(I) \leq k \cdot w(I)$. Combine this with (1) to obtain

$$w(Gr) \geq OPT/k. \tag{2}$$

We now establish a lemma that represents Property 2 of the projection.

LEMMA 3.1. *Let I' be obtained from I via α-good improvements. Then,*

$$\frac{k}{\alpha - 1} \cdot w(I') + proj(I') \geq \frac{k}{\alpha - 1} \cdot w(I) + proj(I).$$

Proof. Let us split OPT into OPT_0, containing those b for which $f_I(b) \in I \cap I'$, and $OPT_1 = OPT - OPT_0$, of those b where $f_I(b) \in I - I'$.

First observe that for $b \in OPT_0$, $w(f_{I'}(b)) \geq w(f_I(b))$, because since b's representative in I exists in I', its representative in I' will be of no less weight. We bound the difference in the projections by

$$\begin{aligned} proj(I) - proj(I') &= \sum_{b \in OPT} w(f_I(b)) - w(f_{I'}(b)) \\ &\leq \sum_{b \in OPT_1} w(f_I(b)) - w(f_{I'}(b)) \\ &\leq \sum_{b \in OPT_1} w(f_I(b)) = \sum_{v \in I - I'} |S_I(v)| w(v) \\ &\leq k \cdot w(I - I'). \end{aligned} \tag{3}$$

Since I' is obtained from $I = I_0$ through α-good improvements $I_1, \ldots, I_n = I'$, we have that

$$\begin{aligned} w(I' - I) &= w(I_n - I_{n-1}) + \cdots + w(I_1 - I_0) \\ &\geq \alpha\, w(I_{n-1} - I_n) + \cdots + \alpha\, w(I_0 - I_1) \\ &= \alpha\, w(I - I'). \end{aligned} \tag{4}$$

Thus, using (4),

$$w(I') - w(I) = w(I' - I) - w(I - I') \geq (\alpha - 1) w(I - I'). \tag{5}$$

Combining (3) with (5) establishes the lemma. ∎

Finally, we establish Property 3, regarding the positive impact of the projection on locally optimal solutions.

LEMMA 3.2. *For an α-locally optimal solution I,*

$$(k+1)w(I) \geq \frac{1}{\alpha} OPT + proj(I).$$

Proof. First note that the set of preimages of elements of a maximal independent set is a partition of the optimal solution. Thus,

$$OPT = \sum_{v \in I} w(S(v)). \tag{6}$$

Since, for any vertex $v \in I$, the independent set $I \cup S(v) - N(S(v))$ is not an α-improvement over I,

$$w(S(v)) < \alpha \cdot w(N_I(S(v))).$$

Adding this up over the vertices in I, applying (6), we have that

$$\frac{1}{\alpha} OPT < \sum_{v \in I} w(N_I(S(v))).$$

The right-hand side can be rewritten as

$$\sum_{v \in I} |\{u \in I : v \in N(S(u))\}| w(v).$$

Each vertex v is adjacent to all of $S(v)$ but to at most k vertices in OPT in total. Thus, v is adjacent to at most $k - |S(v)|$ vertices in $OPT - S(v)$. It follows that

$$\begin{aligned}\frac{1}{\alpha} OPT &< \sum_{v \in I, |S(v)|>0} w(v) + \sum_{v \in I} (k - |S(v)|) w(v) \\ &\leq (k+1)w(I) - proj(I).\end{aligned}$$

∎

The Performance of the Algorithm ANYIMP. Given the properties we have proved for the projection, we are now ready to prove an upper bound on ρ_α, the performance ratio of ANYIMP_α.

THEOREM 3.3. $\rho_\alpha \leq (k + 1 - \frac{1}{\alpha})/(1 + \frac{1}{\alpha} - \frac{1}{\alpha^2})$. *In particular,* $\rho_2 \leq \frac{4k+2}{5}$.

Proof. Any solution I of ANYIMP_α, whether initial, intermediate, or final, satisfies

$$\frac{k}{\alpha - 1} \cdot w(I) + proj(I) \geq \frac{\alpha}{\alpha - 1} OPT. \tag{7}$$

This is true for the initial greedy solution from (1) and (2) and follows inductively for the other solutions from Lemma 3.1. The solution output by ANYIMP_α satisfies the conditions of both Lemma 3.2 and (7). Adding those equations gives

$$\left(k + 1 + \frac{k}{\alpha - 1}\right) \text{ANYIMP}_\alpha \geq \left(\frac{1}{\alpha} + \frac{\alpha}{\alpha - 1}\right) OPT.$$

∎

4. PERFORMANCE ANALYSIS OF THE BESTIMP ALGORITHM

We start by noting that the payoff factors of successive local improvements made by BESTIMP need not be monotonically decreasing. For instance, after performing an improvement, it may become possible to add a vertex, which is an improvement of the infinite payoff. Since this monotonicity property is needed in our analysis, we simulate it as follows.

Let X_i, $i = t, t-1, \ldots, 1$, be the improvements made by the algorithm where the payoff factor drops to a new low. That is, X_t is the first improvement made on the greedy solution; X_{t-1} is the first improvement whose payoff factor is less than that of X_t, etc. Let d_i be the payoff factor of X_i, and define $d_0 = 1$ and $d_{t+1} = d_t$. Let I_{i+1} be the independent set obtained by the algorithm before X_i is applied, and let I_1 be the final solution. Thus, all improvements that lead from I_{i+1} to I_i are d_i-good. Also, I_{i+1} is d_i-locally optimal. Hence, from Lemma 3.2 and (1), we have, for $j = 1, \ldots, t+1$,

$$(k+1)w(I_j) \geq \frac{1}{d_{j-1}} OPT + proj(I_j), \tag{8}$$

$$proj(I_{t+1}) \geq OPT. \tag{9}$$

We analyze the weight of the successively improved solutions using the following potential function.

DEFINITION 4.1. *Let* $\Phi(I, d) = \frac{k+1}{d-1} w(I) + proj(I)$.

Using Lemma 3.1 and that $w(I_i) \geq w(I_{i+1})$, we have, for $i = 1, \ldots, t$, that

$$\Phi(I_i, d_i) \geq \Phi(I_{i+1}, d_i). \tag{10}$$

Consider the function

$$h(i) = \sum_{j=i}^{t-1} \frac{d_i}{d_j} \left(\frac{1}{d_j} - \frac{1}{d_{j+1}} \right) + \frac{d_i}{d_t} \frac{1}{d_t}, \qquad i = 1, \ldots, t.$$

Also, define $h(t+1) = h(t) = 1/d_t$. Observe that h satisfies the recurrence relation

$$h(i) = h(i+1) \frac{d_i}{d_{i+1}} + \frac{1}{d_i} - \frac{1}{d_{i+1}}, \qquad i = 1, \ldots, t.$$

Further, we can use that $1/d_j \geq \frac{1}{x}$, for $x \in [d_j, d_{j+1}]$, to bound $h(i)$ by

$$
\begin{aligned}
h(i) &= d_i \left[\sum_{j=i}^{t-1} \frac{1}{d_j} \left(\int_{d_j}^{d_{j+1}} \frac{1}{x^2} dx \right) + \frac{1}{d_t^2} \right] \\
&\geq d_i \left[\int_{d_i}^{d_t} \frac{1}{x^3} dx + \frac{1}{d_t^2} \right] \\
&= d_i \left(\frac{1}{2d_i^2} - \frac{1}{2d_t^2} + \frac{1}{d_t^2} \right) \\
&> \frac{1}{2d_i}. \qquad (11)
\end{aligned}
$$

LEMMA 4.2. *The following two inequalities hold for $i = 1, 2, \ldots, t+1$.*

$$\Phi(I_i, d_i) \geq \frac{d_i + h(i)}{d_i - 1} OPT \qquad (12)$$

$$(k+1)w(I_i) \geq \left[1 + \frac{h(i)}{d_i} + \frac{d_i - 1}{d_i d_{i-1}} \right] OPT. \qquad (13)$$

Proof. The proof proceeds by induction. Consider the base case, $i = t+1$. Recall that $d_{t+1} = d_t$ and $h(t+1) = h(t) = 1/d_t$. Then, by applying (8) and (9),

$$
\begin{aligned}
\Phi(I_{t+1}, d_{t+1}) &= \frac{k+1}{d_{t+1} - 1} w(I_{t+1}) + proj(I_{t+1}) \\
&\geq \left(\frac{1 + 1/d_t}{d_{t+1} - 1} + 1 \right) OPT \\
&= \frac{d_{t+1} + h(t+1)}{d_{t+1} - 1} OPT.
\end{aligned}
$$

Also, the same equations imply that

$$(k+1)w(I_{t+1}) \geq \left(1 + \frac{1}{d_t} \right) OPT = \left[1 + \frac{h(t+1)}{d_{t+1}} + \frac{d_{t+1} - 1}{d_{t+1} d_t} \right] OPT.$$

Assume by the induction hypothesis that the claim holds for $i = q+1$. We want to show that it also holds for $i = q$. By (10),

$$
\begin{aligned}
\Phi(I_q, d_q) &\geq \Phi(I_{q+1}, d_q) \\
&= \Phi(I_{q+1}, d_{q+1}) + (k+1)w(I_{q+1}) \left(\frac{1}{d_q - 1} - \frac{1}{d_{q+1} - 1} \right).
\end{aligned}
$$

By applying (12) and (13) (with $i = q + 1$) to both terms of the right-hand side, we obtain, after algebraic simplification,

$$\Phi(I_q, d_q) \geq \frac{d_q + h(q)}{d_q - 1} OPT, \tag{14}$$

establishing (12) for the case $i = q$. Furthermore, (14) along with (8) (with $i = q$) gives that

$$\left(1 + \frac{1}{d_q - 1}\right)(k+1)w(I_q) \ \geq \ \left[\frac{1}{d_{q-1}} + \frac{d_q + h(q)}{d_q - 1}\right] OPT,$$

which, when rearranged, establishes (13) for $i = q$. ∎

THEOREM 4.3. *The performance ratio of* BESTIMP *is at most* $2(k+1)/3$.

Proof. By (13) and (11), the final solution I_1 obtained by the algorithm satisfies

$$\begin{aligned}(k+1)w(I_1) &\geq \left[1 + \frac{1}{2d_1^2} + \frac{1}{d_0} - \frac{1}{d_1 d_0}\right] OPT \\ &= \left[2 - \frac{1}{d_1} + \frac{1}{2d_1^2}\right] OPT \geq \frac{3}{2} OPT,\end{aligned}$$

where the last inequality follows from $d_1 \geq 1$. ∎

5. LOWER BOUND CONSTRUCTIONS

The focus of this section is on showing the following lower bound on the performance of BESTIMP.

THEOREM 5.1. *The performance ratio of* BESTIMP *is at least* $(2/3)k - o(k)$.

Construction of the Hard Graph. A bipartite graph (U, V, E) is said to be (l, r)-regular if each vertex in U is of degree l and each vertex in V is of degree r. We need the following result, the proof of which can be found in [5].

Claim 5.2. *For any positive integers* l, r, *and* p, *with* $l \geq r$, *there exists an* (l, r)*-regular bipartite graph on* $(l + r)p$ *vertices that contains a full matching.*

An essential property needed in our construction is the nonexistence of 4-cycles or, in other words, that any pair of vertices shares at most one common neighbor. The following lemma gives a transformation of any graph to a graph without 4-cycles with otherwise identical features.

LEMMA 5.3. *Let G be a graph on n vertices, and let p be a prime number, $p \geq n$. Then, there exists a graph G' containing p copies of each vertex v of G such that*:

1. *If two nodes in G are adjacent, then each copy of one of the nodes is adjacent to exactly one copy of the other node. If the two nodes in G are nonadjacent, their copies are also nonadjacent.*
2. *G' contains no 4-cycle.*

Proof. Denote the vertices of G by $v_1, \ldots, v_n$. The graph G' has vertices $v_{i,x}$, $i = 1, \ldots, n$, $x = 1, \ldots, p$. Vertices $v_{i,x}$ and $v_{j,y}$ are adjacent in G' if v_i and v_j are adjacent in G (and thus $i \neq j$) and if

$$(i+j)^2 + (x+y) \equiv 0 \pmod{p}. \tag{15}$$

Note that this relationship is symmetric, inducing an undirected graph.

For fixed values of i, j, and x, there is a unique value y that satisfies the linear equation (15) over the finite field GF_p. Thus, we see that the construction satisfies the first half of the claim of the lemma. It follows that the degree of a vertex in G is inherited by any of its copies in G'.

To verify the second half of the claim, let $v_{i,x}$ and $v_{j,y}$ be two vertices in G'. Suppose they have a common neighbor $v_{k,q}$. Then, by definition,

$$(i+k)^2 + (x+q) \equiv (j+k)^2 + (y+q) \equiv 0 \pmod{p}.$$

Thus,

$$i^2 + 2ik + x \equiv j^2 + 2jk + y \pmod{p}.$$

or

$$2(i-j)k + (i^2 - j^2 + x - y) \equiv 0 \pmod{p}.$$

Since i, j, x, and y are fixed, this is a linear equation which has exactly one root modulo p (given that $i \neq j$). Thus, there is at most one value between 0 and $n - 1 \leq p - 1$ that satisfies this equation. Hence, any pair of vertices has at most one common neighbor. Thus, G' contains no 4-cycle. ∎

We are now ready to give the main result of this section.

THEOREM 5.4. *For any $k \geq 16$, there is a $k+1$-claw free graph such that*

$$\text{BESTIMP} \leq \left(\frac{3}{2} + \frac{4}{\lfloor k^{1/4} \rfloor}\right) OPT/k.$$

Thus, the performance ratio of BESTIMP *is at least $2k/3 - O(k^{3/4})$.*

Proof. Assume $k \geq 16$. Let $W = \lfloor k^{1/4} \rfloor$ and $t = W^2$. For $i = 0, \ldots, t$, let $\alpha_i = \frac{W+i}{W}$. Thus, $\alpha_0 = 1$ and $\alpha_t = W + 1$. Let $g_i = (1/\alpha_i)(1/\alpha_{i-1} - 1/\alpha_i)$.

We form a graph G_k with vertex sets V, A, B, and C, where A is further partitioned into $A_0, A_1, \ldots, A_t$ and C into $C_1, C_2, \ldots, C_t$.

The intuition behind the construction is as follows: the graph is "almost bipartite" with most of the edges being between V, the optimal independent set, and the other vertices. V has a large number of vertices of large weight. The set of other vertices can be partitioned into two subsets: one subset (A) has a small number of vertices of large weight, while the other subset ($B \cup C$) has a large number of vertices of small weight. The algorithm will first greedily choose A, followed by B. It will then make α_t-improvements, replacing A_t with C_t; followed by α_{t-1}-improvements, replacing A_{t-1} with C_{t-1}; and so on. At the end, the independent set $A_0 \cup B \cup C$ will be locally optimal and hence will represent a possible final solution of BestImp, but its weight will be small compared to $wt(V)$.

The vertex weights are 1 in V and A, $1/k$ in B, and α_i/k in C_i. The relative cardinalities are fixed by the edges specified below.

The edges of the graph are specified as a union of bipartite graphs, by using Claim 5.2. The sets A and V form a $(k, 1)$-regular bipartite graph (i.e., each vertex in A is adjacent to k vertices of V, and each vertex of V is adjacent to one vertex of A), and B and V form a $(k, k/\alpha_t)$-regular bipartite graph (we ignore the rounding in this case for simplicity). For each $i = 1, \ldots, t$, C_i and V form a $(k, \lceil g_i k + 1 \rceil)$-regular bipartite graph, while A_i and C_i are $(k, 1)$-regular. There are no further edges.

These degrees fix the relative sizes of the vertex sets. Namely, $|A| = |V|/k$, $|B| = |V|/\alpha_t$, $|C_i| = (\lceil g_i k + 1 \rceil / k)|V|$, and $|A_i| = |C_i|/k = (\lceil g_i k + 1 \rceil / k^2)|V|$, for $i = 1, \ldots, t$. Finally, $|A_0| = |A| - \sum_{i=1}^{t} |A_i|$.

Note that by appropriately ordering the vertices of A when merging incident bipartite subgraphs, we can ensure the following property: For any vertex x in A_i, there is a perfect matching between its k neighbors in V and its k neighbors in C_i. Applying Lemma 5.3 to the graph constructed, we obtain a graph that contains no 4-cycles, while maintaining the same degree properties, adjacencies of the vertex sets, and the perfect matching property above. This completes the specification of the graph.

To verify that the graph is $k+1$-claw free, note first that vertices of V, B, and A_0 are of degree only k. Vertices of $A - A_0$ are of degree $2k$, but their neighborhoods have independence number at most k since they contain a perfect matching. Similarly, each vertex of C is of degree $k + 1$, but its neighbor in A is adjacent to a neighbor in V.

Execution of BestImp *on the Hard Graph.* We deduce an execution sequence of BestImp on G_k. That means that we specify a poor sequence of choices when several equally good improvements are possible. This is because for an algorithm to attain a given performance ratio, it must do so for any execution sequence. It is, however, easy to modify the weights to insure that only one sequence, which leads to the same lower bound, is possible.

The algorithm initially adds the vertices of A greedily, followed by B, the only remaining nonadjacent vertices. We now observe the following. Denote the vertices of A_i by $a_1^i, \dots, a_q^i$, and the vertices of C_i adjacent to a_t^i by $c_{t,1}^i, c_{t,2}^i, \dots, c_{t,k}^i$. Suppose I consists of the union of B, $\bigcup_{j<i} A_j$, $\{a_1^i, \dots, a_x^i\}$, $\{c_{t,s}^i : t = x+1, \dots, q, s = 1, \dots, k\}$, and $\bigcup_{j>i} C_j$. Then, $I' = (I - \{a_x^i\} \cup \{c_{x,s}^i : s = 1, \dots, k\}$ is an improvement of S of the highest payoff, α_x.

To see that no improvement has a higher payoff, consider any vertex v in I and a subset S of its neighborhood. We split S into two parts, which we bound separately: $S \cap V$ and $S - V$.

Since the graph contains no 4-cycles, any pair of vertices in S shares only v as a neighbor. Thus, we can compare each vertex u in $S \cap V$ with its neighbors in I excluding v. A vertex u in V has by definition at least $g_i k$ neighbors in C_i, for $i = x+1, x+2, \dots, t$, as well as k/α_t neighbors in B. Their combined weight is

$$\frac{1}{\alpha_t} + \sum_{i=x+1}^{t} g(i)k \cdot \frac{\alpha_i}{k} = \frac{1}{\alpha_t} + \sum_{i=x+1}^{t} \left(\frac{1}{\alpha_{i-1}} - \frac{1}{\alpha_i} \right) = \frac{1}{\alpha_x},$$

while the weight of u is 1. That is, the weight of $N_I(u)$ is at least $1/\alpha_x$ times the weight of u.

If $v \in A_0 \cup B$, then $S - V = \emptyset$. If $v \in C$, then S contains at most u from A and no vertex of C. u must have k neighbors in I (including v), of a total weight at least $\alpha_i > 1$, for some $i \geq 1$. If $v \in A$, then $S \cap C$ contains at most k vertices, each of weight α_i, for some $i \leq x$. Thus, $w(S \cap C) \leq k\alpha_x \leq \alpha_x w(v)$. In either case, we have that $w(S - V) \leq \alpha_x w(N_I(S - V))$. Thus we have shown that $(I - S) \cup N_I(S)$ is at most α_x-good. Since $\alpha_0 = 1$, it follows that $ALG = A_0 \cup B \cup C$ is a locally optimal solution. Hence, it is a possible solution obtained by the algorithm.

Weight of the Locally Optimal Solution. The resulting solution, $ALG = A_0 \cup B \cup C$, is locally optimal. The optimal solution is given by V, whose size and weight we denote by OPT. Observe that $|C_i| \leq (g_i + 2/k)OPT$, $|C_i| = k|A_i|$, $w(C_i) = \alpha_i w(A_i) = \alpha_i |A_i|$, and $|A_0| = \frac{1}{k}OPT - \sum_{i=1}^{t} |A_i|$. The weight of ALG is then

$$\begin{aligned} ALG &= |A_0| + |B|\frac{1}{k} + \sum_{i=1}^{t} |A_i|\alpha_i = \frac{1}{k}OPT + |B|\frac{1}{k} + \sum_{i=1}^{t} (\alpha_i - 1)|A_i| \\ &\leq \left[1 + \frac{1}{\alpha_t} + \sum_{i=1}^{t} (\alpha_i - 1)(g_i + 2/k) \right] \frac{OPT}{k} \end{aligned}$$

$$= \left[1 + \frac{1}{\alpha_t} + \sum_{i=1}^{t}\left(\frac{1}{\alpha_{i-1}} - \frac{1}{\alpha_i}\right) - \sum_{i=1}^{t}\frac{1}{\alpha_i}\left(\frac{1}{\alpha_{i-1}} - \frac{1}{\alpha_i}\right) + \sum_{i=1}^{t}(\alpha_i - 1)2/k\right]\frac{OPT}{k}.$$

Recall the telescopic sum $\sum_{i=1}^{t} x_{i-1} - x_i = x_0 - x_t$ for the first sum in the above inequality. Bound the second sum by

$$W^2 \sum_{i=W+1}^{t+W} \frac{1}{i^2(i-1)} \geq W^2 \int_{i=W+1}^{t+W} \frac{1}{x^3} dx \geq \frac{1}{2} - \frac{2}{W}.$$

The last sum is also $\frac{2}{Wk}\sum_{i=1}^{t} i \leq 2t^2/(Wk) \leq 2/W$. Hence, we have that

$$ALG \leq (2 - (1/2 - 2/W) + 2/W)OPT/k = (3/2 + 4/W)OPT/k. \qquad \blacksquare$$

5.1. *Lower bound for* AnyImp

We also have a construction, parametrized by α, which shows that our upper bound for AnyImp$_\alpha$ is asymptotically tight. We restrict our attention to the case when $\alpha \leq \sqrt{k}$.

THEOREM 5.5. *The performance ratio of* AnyImp$_\alpha$ *algorithm is at least*

$$\rho \geq \frac{k-1}{1 + \frac{1}{\alpha} - \frac{1}{\alpha^2} + \frac{\alpha-2}{k}} \geq \frac{4(k-1)}{5} - O(1/k^2).$$

We construct a graph G_α for which AnyImp$_\alpha$ will perform poorly. Let $t = \lceil k/\alpha^2 \rceil + 1$. It consists of the vertex sets A_0, A_1, C, and V of cardinalities $n(k-t)/k(k-1)$, $n(t-1)/k(k-1)$, $n(t-1)/(k-1)$, and n, respectively. The weight of vertices in A_0, A_1, and V is 1, while the weight of vertices in C is α/k. $A_0 \cup C$ and V form a (k, t)-regular bipartite graph and A_1 and C a $(k, 1)$-regular bipartite graph. Additionally, each vertex a of A_1 is adjacent to k distinct vertices of V, and by arguments similar to the previous section we can assume that the neighbors of a in C form a perfect matching with its neighbors in V. This insures that the graph contains no $k+1$-claw. We assume as before, by Lemma 5.3, that the graph contains no 4-cycles. This completes the specification of the graph.

The independent set found by Greedy is $A_0 \cup A_1$, which is maximal. Replacing one by one the vertices of A_1 with their neighborhoods in C is a sequence of α-improvements. Namely, any a in A_1 of weight 1 has k neighbors in C of combined weight α. Those neighbors have no further neighbors in $A_0 \cup A_1 \cup C$, and hence the swap is an α-improvement.

The solution $ALG = A_0 \cup C$ can be seen to be locally α-optimal. Consider a vertex v in ALG, let $B \subseteq N_V(v)$, and let $B' = N_{ALG}(B)$. Since G_α contains no 4 cycles, $|B'| = |B|(t-1)+1$. Thus,

$$w(B') \geq |B'|\alpha/k = (|B|(t-1)+1)\alpha/k \geq \left(\frac{|B|k}{\alpha^2}+1\right)\frac{\alpha}{k} > \frac{|B|}{\alpha}$$
$$= \frac{1}{\alpha}w(B).$$

Hence, B cannot replace B' in an α-improvement, and ALG is a possible solution output by ANYIMP$_\alpha$.

The optimal solution is V, of weight n. The weight of ALG is

$$ALG = \left[\frac{k-t}{k(k-1)} + \frac{(t-1)\alpha}{k(k-1)}\right]n = [1+(t(\alpha-1)-\alpha)/k]n/(k-1).$$

If we now round up the value of t, we obtain

$$ALG \leq [1+((k/\alpha^2+2)(\alpha-1)-\alpha)/k]\frac{n}{k-1} = \left[1+\frac{1}{\alpha}-\frac{1}{\alpha^2}+\frac{\alpha-2}{k}\right]\frac{n}{k-1}.$$

This yields the theorem when rearranged. For large values of k, the ratio is minimized when $\alpha = 2$, for a performance ratio at least $4(k-1)/5$.

6. CONCLUSIONS

Some open issues remain. The time complexity of the algorithms is $O(nk^2\Delta^k)$, where Δ is the maximum degree of the graph. This matches the previous local search algorithms for the weighted set packing problem and holds also for the recent approach of Berman [3]. It would be interesting to determine if a similar performance ratio can be obtained by an algorithm whose time complexity depends less on k, e.g., by $2^{O(k)}n^{O(1)}$. Additionally, it would be interesting to see the proof technique used here applied to other problems.

REFERENCES

1. E. Arkin and R. Hassin, Approximating weighted set packing by local search, *Math. Oper. Res.* **23** (1998), 640–648.
2. V. Bafna, B. Narayan, and R. Ravi, Nonoverlapping local alignments (Weighted independent sets of axis-parallel rectangles), *Discrete Appl. Math.* **71** (1996), 41–53.
3. P. Berman, A $d/2$ approximation for maximum weight independent set in d-claw free graphs, *in* "Proceedings, Seventh Scandinavian Workshop on Algebraic Theory (SWAT)," Lecture Notes in Computer Science, Vol. 1851, pp. 214–219, Springer-Verlag, New York/Berlin, July 2000.

4. B. Chandra and M. M. Halldórsson, Greedy local improvement and weighted set packing approximation, *in* “Proceedings, Tenth ACM-SIAM Symposium on Discrete Algorithms, Jan. 1999,” pp. 169–176.
5. B. Chandra and M. M. Halldórsson, Greedy local improvement and weighted set packing approximation, Res. Rep. RR-31-99, Science Institute, University of Iceland, 1999.
6. M. R. Garey and D. S. Johnson, “Computers and Intractability,” Freeman, New York, 1979.
7. M. M. Halldórsson, Approximating discrete collections via local improvements, *in* “Proceedings Sixth ACM-SIAM Symposium on Discrete Algorithms, Jan. 1995,” pp. 160–169.
8. C. A. J. Hurkens and A. Schrijver, On the size of systems of sets every t of which have an SDR, with an application to the worst-case ratio of heuristics for packing problems, *SIAM J. Discrete Math.* **2**, No. 1 (Feb. 1989), 68–72.
9. D. J. Johnson, C. H. Papadimitriou, and M. Yannakakis, How easy is local search?, *J. Comput. System Sci.* **37** (1988), 79–100.