Adaptive Local Ratio

Julián Mestre *

Abstract

Local Ratio is a well-known paradigm for designing approximation algorithms for combinatorial optimization problems. At a very high level, a local ratio algorithm first decomposes the input weight function w into a positive linear combination of simpler weight functions or *models*. Guided by this process a solution S is constructed such that S is α-approximate with respect to each model used in the decomposition. As a result, S is α-approximate under w as well.

These models usually have a very simple structure that remains "unchanged" throughout the execution of the algorithm. In this work we show that adaptively choosing a model from a richer spectrum of functions can lead to a better local ratio. Indeed, by turning the search for a good model into an optimization problem of its own, we get improved approximations for a data migration problem.

1 Introduction

The Local Ratio technique and the Primal Dual schema are two well-known paradigms for designing approximation algorithms for combinatorial optimization problems. Over the years a clear connection between the two paradigms was observed as researchers found primal-dual interpretations for local-ratio algorithms [11, 4] and vice versa [7, 5]. This culminated with the work of Bar-Yehuda and Rawitz [8] showing their equivalence under a fairly general and encompassing definition of primal-dual and local-ratio algorithms. For a survey of results and a historical account on the Local Ratio technique see [9]; for a survey on the Primal Dual schema see [19, 29].

At a very high level, a local-ratio algorithm consists of two steps. First, the input weight function w is decomposed into a positive linear combination of *models* $\hat{w}_i$, that is, $w = \epsilon_1 \hat{w}_1 + \ldots + \epsilon_n \hat{w}_k$ and $\epsilon_i \geq 0$. Then, guided by this process, a solution S is constructed such that $\hat{w}_i(S) \leq \alpha\, \hat{w}_i(A)$ for any feasible solution A, for all i. We refer to α as the *local ratio* of $\hat{w}_i$. By the Local Ratio Theorem [7], it follows that S is α-approximate with respect to w.

Typically the models used in local-ratio approximation algorithms are 0-1 functions or simple aggregates of structural features of the problem at hand. (In the primal-dual parlance this corresponds to increasing some dual variables uniformly when constructing the dual solution.) Furthermore, the structure of the models remains "unchanged" throughout the execution of the algorithm. For example, consider the vertex cover problem. Bar-Yehuda and Even [7] set the weight of the endpoints of a yet-uncovered edge to 1 and the remaining vertices to 0; Clarkson [12] chooses a set of yet-uncovered edges forming a star and sets the weight of each vertex to the number star edges incident on it; while Gandhi et al. [17] and Bar-Yehuda [6] set the weight of each vertex to the number of yet-uncovered edges incident on it.

This paper studies a problem for which adaptively choosing a model from a richer spectrum of weight functions leads to a better local ratio, and thus to better approximations. Indeed, by turning the search for a good model into an optimization problem of its own, we get improved approximations for a *data migration* problem. We hope that our findings encourage the study of non-uniform updates for local-ratio or primal-dual algorithms; perhaps in some cases, as in our problem, this may help realize the full potential of these techniques.

1.1 Our results The *data migration* problem arises in large storage systems, such as *Storage Area Networks* [23], where a dedicated network of disks is used to store multimedia data. As the data access pattern evolves over time, the load across the disks must be rebalanced so as to continue providing efficient service. This is done by computing a new data layout and then *migrating* data to convert the current data layout to the target data layout. While migration is being performed, the storage system is running sub-optimally, so it is important to perform the migration quickly. The problem can be modeled [24] with a *transfer multigraph* (V, E) where each vertex corresponds to a disk and each edge $(u, v) \in E$ corresponds to a data item that must be transferred between u and v. A disk can be involved

*Max-Planck-Institute für Informatik, Saarbrücken, Germany. Email: `jmestre@mpi-inf.mpg.de`. Research done while at the University of Maryland, and supported by the Ann G. Wylie Dissertation Fellowship.

in at most one transfer at a time and each data transfer takes one unit of time. We are to schedule the transfers so as to minimize the sum of completion time of the disks. The problem is NP-hard but 3-approximations are known [24, 15].

First, we cast the primal-dual algorithm Gandhi and Mestre [15] as a local-ratio algorithm and provide a family of instances showing their analysis is tight. To overcome these difficult instances we propose to adaptively choose a model minimizing the local ratio and formulate the problem of finding such a model as a linear program. Interestingly, our algorithm is neither purely combinatorial nor LP rounding, but lies somewhere in between. Every time the weight function needs to be updated, an LP is solved to find the best model. These LPs are much smaller that the usual LP formulations, so the our scheme should be faster than an LP rounding algorithm.

In the analysis we show that the models found using the LP exhibit a better local ratio than the usual 0-1 models. Somewhat surprisingly a precise characterization of the local ratio can be derived analytically. We prove that the overall scheme is a $1 + \phi$ approximation, where $\phi = \frac{1+\sqrt{5}}{2}$ is the Golden ratio, and give a family of instances achieving this ratio.

To derive the worst-case local ratio of our scheme we use a method similar to the factor-revealing LP approach of Jain et al. [22], which has been successfully applied in the analysis of many greedy heuristics [22, 2, 21, 10]. The idea there is to formulate as an LP the problem of building a worst-case instance maximizing the approximation ratio achieved by the heuristic at hand, and then upperbound the cost of this LP by producing a dual solution. We too formulate as a mathematical program the problem of building a worst-case instance maximizing the local ratio. The main difference here is that, since we already use an LP to guide our local-ratio algorithm, the resulting factor-revealing program is non-linear. Even though we cannot solve this program numerically, we are still able to prove a tight bound of $1 + \phi$ on its cost.

1.2 Related work on data migration

As noted by Coffman et al. [13], if every transfer takes one unit of time a schedule is simply a proper edge coloring of the transfer multigraph.

Many objective functions have been studied within this framework such as minimizing the maximum disk completion time (makespan) [13, 27], sum of disk completion times [24, 18, 15] and sum of transfer completion times [3, 24, 26, 30]. A common variant is to allow each transfer to have an arbitrary processing time [28, 24, 13, 18, 16]. Generalizations of the makespan minimization problem in which there are storage constraints on disks or constraints on how the data can be transferred have also been studied [20, 1, 23, 25].

This paper is mainly concerned with the objective of minimizing the weighted sum of vertex completion times. Kim [24] proved that the problem is NP-hard and showed that a simple greedy algorithm guided by an optimal LP solution is a 3-approximation. Gandhi et al. [18] showed that Kim's analysis is tight. Finally, Gandhi and Mestre [15] showed that the same greedy algorithm guided by a dual update is also a 3-approximation, but that an arbitrary greedy schedule can be a $\omega(1)$-factor away from optimum.

2 Algorithmic Framework

The input to our problem consists of a transfer graph $G = (V, E)$ and a weight function $w : V \to R^+$. For ease of exposition, we assume for now that G is simple; later, in Section 5 we will show how to remove this assumption. A feasible schedule $S : E \to Z^+$ is a proper edge-coloring of G, that is, if two edges $e_1 \neq e_2$ are incident on the same vertex then $S(e_1) \neq S(e_2)$. We are to find a scheduling minimizing $w(S) = \sum_{u \in V} w(u) \max_{v \in N(u)} \{S(u, v)\}$.

Let us cast the primal-dual algorithm of Gandhi and Mestre [15] as a local-ratio algorithm, and in the process, we generalize it slightly. Algorithm ALR has two stages: labeling and scheduling. The labeling stage assigns a label ℓ_u to every $u \in V$; these labels are then used to guide the scheduling stage. The pseudo-code of ALR is given in Figure 1.

Initially every node is unlabeled, i.e., $\ell_v = \mathbf{nil}$ for all $v \in V$. Denote the set of unlabeled neighbors of u with $\mathrm{UN}(u) = \{v \in \mathrm{N}(u) \mid \ell_v = \mathbf{nil}\}$. In each iteration, choose a node u with the maximum number of unlabeled neighbors $\Delta = |\mathrm{UN}(u)|$. Then choose a *model* $\hat{w} : V \to R^+$ with support in $\mathrm{UN}(u)$, find the largest $\epsilon \geq 0$ such that $\epsilon\hat{w} \leq w$, and set $w \leftarrow w - \epsilon\hat{w}$. As a result, at least one vertex in $\mathrm{UN}(u)$ has zero weight in the updated w; set the label of these vertices to Δ. This continues until all vertices are labeled and $w = 0$.

Once the labels are computed, the edges $(u, v) \in E$ are sorted in increasing value of $\min\{\ell_u, \ell_v\}$, breaking ties with $\max\{\ell_u, \ell_v\}$. The edges are scheduled greedily in sorted order: Start with the empty schedule, and add the edges, one by one in sorted order, to the current schedule as early as possible without creating a conflict.

The labels guide the scheduling phase and let us bound the finishing time of the vertices. To motivate the use of the labels Appendix A shows that the same scheduling procedure, when guided by the degrees instead of the labels can yield a solution with cost $\omega(1)$ times the optimum.

```
ALR$(V, E, w)$
  1  // LABELING STAGE
  2  **for** $u \in V$ **do** $\ell_u \leftarrow$ **nil**
  3  **repeat**
  4      choose $u \in V$ maximizing $\Delta = |\mathrm{UN}(u)|$
  5      choose $\hat{w}$ with support in $\mathrm{UN}(u)$
  6      $w \leftarrow w - \min\left\{ \frac{w(u)}{\hat{w}(u)} \,\middle|\, \hat{w}(u) > 0 \right\} \hat{w}$
  7      **for** $v \in \mathrm{UN}(u) \,|\, w(v) = 0$ **do** $\ell_v \leftarrow \Delta$
  8  **until** every vertex is labeled
  9  // SCHEDULING STAGE
 10  sort $(u, v) \in E$ in increasing
     value of $\langle \min\{\ell_u, \ell_v\}, \max\{\ell_u, \ell_v\} \rangle$
 11  $S \leftarrow$ emtpy schedule
 12  **for** $e \in E$ in sorted order **do**
 13      add $e$ to $S$ as early as possible
 14  **return** $S$
```

Figure 1: Pseudo-code for ALR.

LEMMA 2.1. ([15]) *In S, every vertex $v \in V$ finishes no later than $\ell_v + d_v - 1$.*

Proof. Let (v, w) be the edge incident on v that is scheduled the latest in S. Note that there are at most $d_v - 1$ edges incident on v that come before (v, w) in sorted order. How many edges incident on w are there before (v, w) in sorted order? We claim that at most $\ell_v - 1$. Thus, (v, w) must be scheduled not later than $\ell_v + d_v - 1$. To prove our claim, let $X = \{x \in \mathrm{N}(w) \,|\, \ell_x \leq \ell_v\}$. The quantity we want to bound is clearly at most $|X| - 1$. Notice that the value of the labels assigned in Line 7 of ALR can only decrease with time. Consider the first iteration of the algorithm in which the node u chosen in Line 4 of ALR was such that $|\mathrm{UN}(u)| = \ell_v$; at this point in time $\mathrm{UN}(w) = X$. Since u is chosen to maximize $|\mathrm{UN}(u)|$ it follows that $|X| = |\mathrm{UN}(w)| \leq |\mathrm{UN}(u)| = \ell_v$. □

As it stands, the algorithm is underspecified: We have not described how the model $\hat{w}$ chosen in Line 6. It is important to realize though, that Lemma 2.1 holds regardless of our choice of $\hat{w}$. Gandhi and Mestre [15] proposed using $\hat{w}(v) = \mathbb{I}[v \in \mathrm{UN}(u)]$ as a model, where $\mathbb{I}[\cdot]$ is a 0-1 indicator function, and showed that this is a 3-approximation. In order to gain some intuition let us show that the local ratio of $\hat{w}(v) = \mathbb{I}[v \in \mathrm{UN}(u)]$ is at most 3.

LEMMA 2.2. *If Line 6 always uses $\hat{w}(v) = \mathbb{I}[v \in \mathrm{UN}(u)]$ then $\hat{w}(S) \leq 3\,\hat{w}(A)$ for any schedule A.*

Proof. An obvious lowerbound on $\hat{w}(A)$ is $\sum_{v \in V} \hat{w}(v)\, d_v = \sum_{v \in \mathrm{UN}(u)} d_v$. Furthermore, since nodes in $\mathrm{UN}(u)$ share u as a common neighbor, it follows that $\hat{w}(A) \geq \sum_{i \in [\Delta]} i > \Delta^2/2$, where $\Delta = |\mathrm{UN}(u)|$. On the other hand, by Lemma 2.1

$$\begin{aligned} \hat{w}(S) &\leq \sum_{v \in \mathrm{UN}(u)} (\ell_v + d_v - 1), \\ &\leq \Delta^2 + \sum_{v \in \mathrm{UN}(u)} d_v, \\ &< 3\,\hat{w}(A). \end{aligned}$$

The second inequality follows from the fact that the labels that ALR assigns only decrease with time. □

Since S is a 3-approximation with respect to every model and the input weight function w is a positive linear combination of these models, it follows that S is 3-approximate with respect to w as well. It is worth pointing out that the bound on the local ratio of Lemma 2.2 is essentially tight with respect to Lemma 2.1. To see this, consider what happens when the ith vertex in $\mathrm{UN}(u)$ has degree $d_i = i$. Of course, this alone does not imply a tight bound on the overall approximation guarantee, but as we will see in Lemma 3.2, there is a family of instances where the algorithm in [15] produces a schedule whose cost is $3 - o(1)$ times optimum.

Notice that the degree sequence $d_i = i$ can be easily circumvented if we use a different model: Instead of $\hat{w}(v) = \mathbb{I}[v \in \mathrm{UN}(u)]$, which has a local ratio ≈ 3, use the model that just sets to 1 the node in $\mathrm{UN}(u)$ with degree Δ, which in this case has a local ratio of ≈ 2. Indeed, in general choosing the best model between $\hat{w}(v) = \mathbb{I}[v \in \mathrm{UN}(u)]$ and $\hat{w}(v) = \mathbb{I}[v \in \mathrm{argmax}_{x \in \mathrm{UN}(u)} d_x]$ leads to a local ratio of $3 - \beta^2 \approx 2.802$, where $\beta \approx 0.445$ is a root of the polynomial $p(x) = x^3 - x^2 - 2x + 1$. The ratio can be achieved with with the degree sequence $d_i = \min\{i, \lceil \beta\Delta \rceil\}$, which can be shown to be tight using Lemma 3.2.

Besides a modest improvement in the approximation guarantee, this suggests a general line of attack: In each iteration find a model that minimizes the local ratio.

3 Minimizing the local ratio

The abstract problem we are to solve is, given a sequence $d = (d_1, d_2, \ldots, d_\Delta)$ corresponding to the degrees of vertices in $\mathrm{UN}(u)$, find weights $\hat{w} = (\hat{w}_1, \hat{w}_2, \ldots, \hat{w}_\Delta)$ minimizing the local ratio of $\hat{w}$.

In order to evaluate the goodness of a given model $\hat{w}$ we first need an upper bound on $\hat{w}(S)$, where S is the schedule produced by ALR. For this we use Lemma 2.1

and the fact that the values of labels assigned in Line 7 of ALR can only decrease with time.

DEFINITION 1. $\mathrm{UB}(d,\hat{w}) = \sum_{i\in[\Delta]} \hat{w}_i\,(d_i + \Delta - 1)$.

Similarly, we need a lower bound on $\hat{w}(A)$, where A can be any schedule. Note that A must schedule the edges from $\mathrm{UN}(u)$ to u at different time slots; this induces a total order on $\mathrm{UN}(u)$, which we denote by the permutation $\sigma : [\Delta] \to [\Delta]$. Notice that vertex i cannot finish earlier than $\sigma(i)$ since (u,i) is the $\sigma(i)$th edge incident on u to be scheduled, nor earlier than d_i since all edges incident on i must be scheduled before it finishes.

DEFINITION 2. $\mathrm{LB}(d,\hat{w}) = \min\limits_{\sigma:[\Delta]\to[\Delta]} \sum\limits_{i\in[\Delta]} \hat{w}_i \max\{d_i,\sigma(i)\}$.

It follows from the above discussion that $\hat{w}(S) \le \mathrm{UB}(d,\hat{w})$ and $\mathrm{LB}(d,\hat{w}) \le \hat{w}(A)$ for all A. Hence, the minimum local ratio for d can be expressed as a function of UB and LB.

DEFINITION 3. *Let* $\rho(d) = \inf\limits_{\hat{w}} \dfrac{\mathrm{UB}(d,\hat{w})}{\mathrm{LB}(d,\hat{w})}$ *be the minimum local ratio of* d.

We now turn our attention to the problem of computing a model $\hat{w}$ with a local ratio that achieves $\rho(d)$. This can be done using the program $\min\{\mathrm{UB}(d,\hat{w}) \,|\, \mathrm{LB}(d,\hat{w}) \ge 1, \hat{w} \ge 0\}$, which can be rewritten as a linear program.

$$\text{(LP1)} \qquad \min \sum_{i\in[\Delta]} (d_i + \Delta - 1)\,\hat{w}_i$$

subject to

$$\text{(3.1)} \qquad \sum_{i\in[\Delta]} \hat{w}_i \max\{d_i,\sigma(i)\} \ge 1 \qquad \forall \sigma : [\Delta] \to [\Delta]$$

$$\text{(3.2)} \qquad \hat{w}_i \ge 0 \qquad \forall\, i \in [\Delta]$$

Clearly, LP1 computes a model $\hat{w}$ with local ratio $\rho(d)$. Even though LP1 is exponentially large, it can be solved in polynomial time using the ellipsoid method—the separation oracle involves solving a minimum assignment problem. The ellipsoid method, however, is not practical, so below we derive a more succinct formulation.

$$\text{(LP2)} \qquad \min \sum_{i\in[\Delta]} (d_i + \Delta - 1)\,\hat{w}_i$$

subject to

$$\text{(3.3)} \qquad \sum_{i\in[\Delta]} (y_i - z_i) \ge 1$$

$$\text{(3.4)} \qquad y_i - z_j \le \max(d_i, j)\,\hat{w}_i \qquad \forall\, i,j \in [\Delta]$$

$$\text{(3.5)} \qquad y_i, z_i, \hat{w}_i \ge 0 \qquad \forall\, i \in [\Delta]$$

The idea behind LP2 is to replace the cost of the assignments in (3.1) with their dual lower bound. If $(\hat{w}, y, z)$ is a feasible solution of LP2 then $\hat{w}$ is a feasible solution of LP1, since for any assignment σ,

$$\begin{aligned} 1 &\le \sum_{i\in[\Delta]} (y_i - z_i), \\ &= \sum_{i\in[\Delta]} (y_i - z_{\sigma(i)}), \\ &\le \sum_{i\in[\Delta]} \hat{w}_i \max\{d_i,\sigma(i)\}. \end{aligned}$$

Since the polytope of the assignment problem is integral [14, Chapter 5.3], the converse also holds. In other words, for a fixed $\hat{w}$, there exist y and z obeying (3.4) such that

$$\min_{\sigma} \sum_{i\in[\Delta]} \hat{w}_i \max\{d_i,\sigma(i)\} = \sum_{i\in[\Delta]} (y_i - z_i).$$

This finishes the description of ALR from Figure 1. Namely, in each iteration of the labeling stage, Line 6 must solve LP2 to find the best model for the degree sequence of nodes in $\mathrm{UN}(u)$.

DEFINITION 4. *Let* $\rho = \sup_d \rho(d)$ *and* $\rho_\Delta = \max_{d:|d|=\Delta} \rho(d)$.

THEOREM 3.1. ALR *is a* ρ*-approximation for the data migration problem and this is tight.*

The proof that the algorithm is a ρ-approximation is similar to that of Lemma 2.2. Let S be the schedule produced by ALR. For every model $\hat{w}$ used by ALR, by definition, S is ρ-approximate with respect to $\hat{w}$. Since the input weight function is a linear combination of these models S is a ρ-approximation. The tightness claim follows from the next two lemmas.

LEMMA 3.1. *For any* Δ*, we have* $\rho_\Delta < \rho_{2\Delta}$.

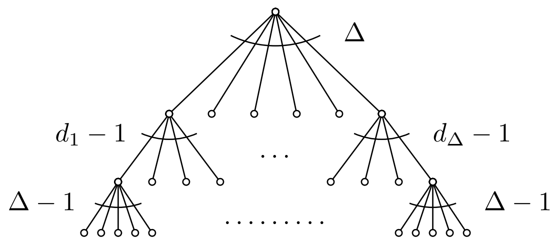


Figure 2: Tight instance for ALR.

Proof. Let d be such that $\rho_\Delta = \rho(d)$. Then define $d'_{2i-1} = d'_{2i} = 2d_i$. Let $(\hat{w}', y', z')$ be a solution LP2 for d' with cost $\rho(d')$. Define $y_i = y'_{2i-1} + y'_{2i}$, $z_i = z'_{2i-1} + z'_{2i}$ and $\hat{w}_i = 2(\hat{w}'_{2i-1} + \hat{w}'_{2i})$. Then $(\hat{w}, y, z)$ is a feasible solution for d with cost less than $\rho(d')$. Thus $\rho_\Delta = \rho(d) < \rho(d') \leq \rho_{2\Delta}$. □

Hence, for our lower bound we only need to worry about large values of Δ. Let d be a degree sequence of length Δ such that $\rho_\Delta = \rho(d)$. For technical reasons, assume that $d_i < \Delta$ (otherwise the local ratio is less than 2) and $d_i > 1$ for $i \geq 2$ (otherwise we can use a stronger upper bound). We show that for large Δ, the algorithm can produce solutions arbitrarily close to ρ_Δ.

LEMMA 3.2. *Suppose that Line 6 of* ALR *always chooses the model $\hat{w}$ when the vertice in $\mathrm{UN}(u)$ have degrees $d_1, \ldots, d_\Delta$, where $d_i > 1$ for $i \geq 2$ and $d_i < \Delta$ for all i. Then the algorithm can produce a schedule with cost $\frac{\mathrm{UB}(d,w)}{\mathrm{LB}(d,w)}\left(1 - \frac{1}{\Delta}\right)$ times the optimum.*

Proof. Consider the instance in Figure 2, namely, a tree with four levels. The ith node in the second level has weight of $\hat{w}_i$, nodes in other levels have weight zero. The root has degree Δ; the ith node in the second level has degree d_i, i.e., it has $d_i - 1$ children; nodes in the third level have degree Δ, i.e., each node has $\Delta - 1$ children.

Consider an execution of the algorithm that chooses the root in the first iteration, as a result, all nodes in the second level get a label of Δ. In the next $\sum_i (d_i - 1)$ iterations the leaves are labeled $\Delta - 1$. Finally, the remaining nodes get a label less than $\Delta - 1$. Now consider a node in the third level, note that the children are labeled $\Delta - 1$, while its parent is labeled Δ. Therefore, the edges between the third and forth level will be scheduled before those between the second and the third level. Also assume that the edges incident on the root are scheduled from right to left. As a result, the ith node in the second level finishes by $d_i + \Delta - 2$. On the other hand, the optimal solution has cost precisely $\mathrm{LB}(d, \hat{w})$. Therefore, the approximation ratio is precisely

$$\frac{\sum_{i\in[\Delta]} \hat{w}_i\,(d_i + \Delta - 2)}{\mathrm{LB}(d,\hat{w})} = \frac{\mathrm{UB}(d,\hat{w}) - \sum_{i\in[\Delta]} \hat{w}_i}{\mathrm{LB}(d,\hat{w})}, \geq \alpha\left(1 - \frac{1}{\Delta}\right).$$

□

As a corollary, we get that the analysis of Gandhi and Mestre [15] is essentially tight since the model $\hat{w}_i = 1$ has a local ratio of $3 - \frac{2}{\Delta+1}$ for the degree sequence $d_i = i$.

However, if we use LP2 to find the best model then the approximation factor becomes ρ, which by Lemmas 3.1 and 3.2 is tight. It only remains to bound ρ. Somewhat surprisingly, a precise characterization in terms of the Golden ratio $\phi = \frac{1+\sqrt{5}}{2} \approx 1.618$ can be derived analytically.

THEOREM 3.2. $\rho = 1 + \phi$.

The next section is devoted to proving this theorem. Figure 3 shows ρ_Δ for small values of Δ obtained through exhaustive search. The MATLAB code for computing the best model for a given degree sequence is available from the author's website.

4 A tight bound for ρ

We start by showing that $\rho \leq 1 + \phi$. In a sense, we need to argue that every degree sequence d has a good model. Recall that the best model can be found with the linear program LP2. At first glance this may seem like an obstacle since we are essentially treating LP2 as a black box; however, we can exploit linear duality to show the upper bound.

The idea is to replace LP2 with its dual problem LP3 given below. By the Strong Duality Theorem the optimal solution of LP2 and LP3 have the same cost.

(LP3) $$\max\ \alpha$$

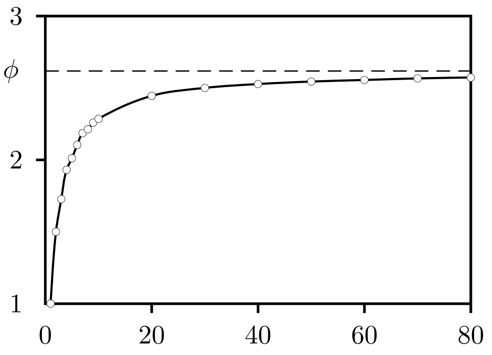
subject to

$$\sum_{j\in[\Delta]} x_{ij} \geq \alpha \qquad \forall\, i \in [\Delta] \tag{4.6}$$

$$\sum_{i\in[\Delta]} x_{ij} \leq \alpha \qquad \forall\, j \in [\Delta] \tag{4.7}$$

$$\sum_{j\in[\Delta]} \max(d_i, j)\, x_{ij} \leq d_i + \Delta - 1 \qquad \forall\, i \in [\Delta] \tag{4.8}$$

$$x_{ij}, \alpha \geq 0 \qquad \forall\, i, j \in [\Delta] \tag{4.9}$$



Δ	ρ_Δ
1	1
2	1.5
3	1.7273
4	1.9310
5	2.0115
6	2.1042
7	2.1863
8	2.2129
9	2.2589
10	2.2857

Δ	ρ_Δ
20	2.4453
30	2.5006
40	2.5275
50	2.5447
60	2.5556
70	2.5667
80	2.5728
$\vdots$	$\vdots$
∞	$1+\phi$

Figure 3: Experimental evaluation of $\rho_\Delta = \rho_{d:|d|=\Delta}\rho(d)$.

Recall that $\rho_\Delta = \max_{d:|d|=\Delta} \rho(d)$. Suppose we modify LP3 by letting $d_1, \ldots, d_\Delta$ be variables in $[\Delta]$. The result is a mathematical program for ρ_Δ, albeit a non-linear one. Denote this program by NLP$_\Delta$. The plan is to show that NLP$_\Delta$ is upperbounded by $1+\phi$. To that end, let us first derive some structural properties about the solutions for NLP$_\Delta$.

Lemma 4.1. *There is an optimal solution (x, d, α) for* NLP$_\Delta$ *such that for all i:*

i) $d_i = \min\{k \mid \sum_{j=1}^k x_{i,j} \geq 1\}$,

ii) $x_{i,j} = 0$ *for all* $j \leq d_i - 2$, *and*

iii) If $x_{i,d_i-1} \neq 0$ *then* $x_{k,d_i} = 0$ *for all* $k < i$.

Proof. Suppose $d_i < \min\{k \mid \sum_{j=1}^k x_{i,j} \geq 1\}$. If d_i obeys (4.8) then we can increment d_i by 1, since this increases the left hand side of (4.8) by less than 1 and its right hand side by 1. Similary if $d_i > \min\{k \mid \sum_{j=1}^k x_{i,j} \geq 1\}$ we can safely decrease d_i without violating feasibility. Thus, from now on we can assume that i) is always satisfied.

First sort the rows of x so that $d_1 \leq d_2 \leq \ldots \leq d_\Delta$. The plan is to modify x row by row until ii) and iii) are satisfied. For the base case $i = \Delta$. Suppose there exists $k < d_i$ such that $x_{i,k} > 0$. That means $x_{i,d_i} < \alpha$. By (4.8) and the fact that $i = \Delta$, there must be an $i' < i$ such that $x_{i',d_i} > 0$. Let $\epsilon = \min\{x_{i,k}, x_{i',d_i}\}$. Decrease $x_{i,k}$ and x_{i',d_i} by ϵ, and increase x_{i,d_i} and $x_{i',k}$ by ϵ. Note that the update does not affect constraint (4.8) for i or i'. However, it may decrease $d_{i'}$ (as function of x) in which case we must resort the rows. The update is repeated until $x_{i,k} = 0$ for all $k < d_i$.

Assuming that rows in $[i+1, \Delta]$ obey ii) and iii), we show how to modify the ith row. The idea is very similar to the base case, the only difference is that if there exists $k < d_i$ such that $x_{i,k} > 0$ then the fact that $x_{i,d_i} < \alpha$ is not enough to conclude the existance of $i' < i$ such that $x_{i',d_i} > 0$ since the remaining non-zero entries in the d_ith column may be in rows $i' > i$. However, if this is the case then by iii) we have $x_{i',d_i-1} = 0$ for $i' > i$ since $x_{i,d_i} > 0$. Thus, we can safely do the update for x_{i,d_i-1} until $x_{i,k} = 0$ for all $k \leq d_i - 2$. Also note that if we have to switch from x_{i,d_i} to x_{i,d_i-1} then $x_{i',d_i} = 0$ for $i' < i$. Putting all together we get properties ii) and iii) for the ith row. □

Lemma 4.2. *For any Δ, the objective value of* NLP$_\Delta$ *is upperbounded by* $(1+\phi) + \frac{1}{\Delta-1}$.

Proof. Let (x, d, α) be a solution for NLP$_\Delta$satisfying Lemma 4.1. Our goal is to show that $\alpha \leq 1 + \phi + \frac{1}{\Delta-1}$.

Let k be the largest index such that $(\phi-1)k < d_k$. If $k = \Delta$ then by constraint (4.8) we have

$$\sum_{j\in[\Delta]} \max\{d_\Delta, j\}\, x_{i,j} \leq d_\Delta + \Delta - 1,$$

and therefore,

$$\alpha\, d_\Delta \leq d_\Delta + \Delta. \tag{4.10}$$

Where (4.10) follows from (4.7). Rearranging the terms in (4.10) we get

$$\alpha \leq \frac{\Delta}{d_\Delta} + 1 < \frac{\Delta}{(\phi-1)\Delta} + 1 = \phi + 1.$$

Let us consider the case where $k < \Delta$. Adding up constraints (4.8) for all i such that $k < i \leq \Delta$ we get

$$\sum_{\substack{i=k+1\\ j=1}}^{\Delta} \max\{d_i, j\}\, x_{i,j} \leq \sum_{i=k+1}^{\Delta} (d_i + \Delta - 1),$$

and therefore,

$$\sum_{j=d_k}^{d_k+\Delta-k-1} \alpha\, j \leq \sum_{i=k+1}^{\Delta} (\phi-1)i + (\Delta-k)(\Delta-1). \tag{4.11}$$

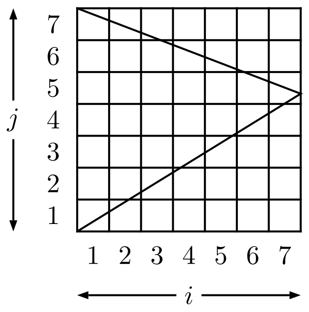


Figure 4: How to construct a solution for NLP_Δ.

Where (4.11) follows from the fact that $x_{i,j} = 0$ for all $i > k$ and $j < d_k$ (by Lemma 4.1), every row and column of x add up to α (by constraints (4.6) and (4.7)), and $d_i \leq (\phi-1)i$ for all $i > k$ (by definition of k). Simplfying (4.11) using $d_k > (\phi-1)k$, we get

$$\alpha \leq \frac{(\phi-1)(k+1+\Delta)+2(\Delta-1)}{2(\phi-1)k+\Delta-k-1}.$$

The right hand side of (4) is maximized when $k = \Delta - 1$. Thus,

$$\begin{aligned}\alpha &\leq \frac{(\phi-1)2\Delta+2(\Delta-1)}{2(\phi-1)(\Delta-1)}\\ &= \frac{2(\phi\Delta-1)}{2(\phi-1)(\Delta-1)}\\ &= (1+\phi)+\frac{1}{\Delta-1}.\end{aligned}$$

Precisely what we needed. □

By Lemmas 3.1 and 4.2, we get that $\rho \leq 1+\phi$. The next lemma finishes the proof of Theorem 3.2 by showing that $\rho \geq 1+\phi$.

LEMMA 4.3. *For every Δ, the objective value of NLP_Δ is lowerbounded by $(1+\phi)\left(1-\frac{3}{\Delta}\right)$.*

Proof. The plan is to construct a feasible solution (x, d, α) for NLP_Δ with $\alpha = (1+\phi)\left(1-\frac{3}{\Delta}\right)$. Since the cost of the optimal solution can only be larger than this, the lemma follows.

Imagine drawing on the Cartesian plane a Δ by Δ grid, and lines $l_1 = (\phi-1)x$ and $l_2 = \Delta-(2-\phi)x$. Figure 4 shows the grid for $\Delta = 7$. Define the cell (i,j) to be the square $[i-1,i]\times[j-1,j]$. Suppose the intersection of cell (i,j) and l_1 defines a segment of length L then we set $x_{i,j} = c\,L\,\frac{\phi}{\sqrt{3-\phi}}$, where c is a constant, which will be choosen shortly to make the solution feasible. Similarly, if the intersection of cell (i,j) and l_2 has length L then we set $x_{i,j} = c\,L\,\frac{1}{\sqrt{6-3\phi}}$. It is easy to check that for every j and i:

$$\sum_{j'\in[\Delta]} x_{i,j'} = \sum_{i'\in[\Delta]} x_{i',j} = c\,(1+\phi).$$

Hence, due to constraints (4.6) and (4.7), the cost of the dual solution is $\alpha = c\,(1+\phi)$. Let $d_i = \lceil(\phi-1)i\rceil$. It only remains to find the largest c that satisfying (4.8). To that end we ask that for all $i \in [\Delta]$

$$\lceil(\phi-1)i\rceil\,c\,\phi + \lceil\Delta-(2-\phi)(i-1)\rceil\,c \leq \lceil(\phi-1)i\rceil+\Delta-1.$$

It can be verified that the above inequality holds for $c = 1-\frac{3}{\Delta}$. □

5 Generalizations

Throughout the paper we have assumed that the transfer graph G is simple. In practice G is typically a multigraph, so we now show how to modify the description of ALR given in Figure 1 to handle multigraphs. Let $E(u,v)$ denote the set of edges between u and v. First, in Line 4 of ALR we choose a vertex u maximizing $\Delta = \sum_{v\in\mathrm{UN}(u)}|E(u,v)|$. To compute the model $\hat{w}$ we create a degree sequence $d'_1,\ldots d'_\Delta$ by making $|E(u,v)|$ copies of d_v for each $v \in N(u)$. Solve LP2 to get weights $\hat{w}'$, and then set $\hat{w}(v)$ to be the sum of $\hat{w}'_i$ for the indices i induced by d_v. These two modifications make the discussion on the approximation ratio for ALR to carry over to multigraphs.

Acknowledgements: Thanks to Rajiv Gandhi and Dror Rawitz for helpful discussions. Special thanks to Reuven Bar-Yehuda for explaining the Local Ratio technique.

References

[1] E. Anderson, J. Hall, J. Hartline, M. Hobbes, A. Karlin, J. Saia, R. Swaminathan, and J. Wilkes. An experimental study of data migration algorithms. In *Proceedings of the 5th Workshop on Algorithm Engineering (WAE),*, 2001.

[2] N. Bansal, L. Fleischer, T. Kimbrel, M. Mahdian, B. Schieber, and M. Sviridenko. Further improvements in competitive guarantees for QoS buffering. In *Proceedings of the 14th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 196–207, 2004.

[3] A. Bar-Noy, M. Bellare, M. M. Halldórsson, H. Shachnai, and T. Tamir. On chromatic sums and distributed resource allocation. *Information and Computation*, 140: 183–202, 1998.

[4] A. Bar-Noy, R. Bar-Yehuda, A. Freund, J. S. Naor, and B. Schieber. A unified approach to approximating resource allocation and scheduling. *Journal of the ACM*, 48(5):1069–1090, 2001.

[5] R. Bar-Yehuda. One for the price of two: A unified approach for approximating covering problems. *Algorithmica*, 27(2):131–144, 2000.

[6] R. Bar-Yehuda. Using homogeneous weights for approximating the partial cover problem. *Journal of Algorithms*, 39(2):137–144, 2001.

[7] R. Bar-Yehuda and S. Even. A local-ratio theorem for approximating the weighted vertex cover problem. *Annals of Discrete Mathematics*, 25:27–46, 1985.

[8] R. Bar-Yehuda and D. Rawitz. On the equivalence between the primal-dual schema and the local ratio technique. *SIAM Journal on Discrete Mathematics*, 19(3): 762–797, 2005.

[9] R. Bar-Yehuda, K. Bendel, A. Freund, and D. Rawitz. Local ratio: A unified framework for approximation algorithms. *ACM Computing Surveys*, 36(4):422–463, 2004.

[10] B. E. Birnbaum and K. J. Goldman. An improved analysis for a greedy remote-clique algorithm using factor-revealing LPs. In *Proceedings of the 9th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, pages 49–60, 2006.

[11] F. Chudak, M. X. Goemans, and D. S. Hochbaum. A primal-dual interpretation of recent 2-approximation algorithms for the feedback vertex set problem in undirected graphs. *Operations Research Letter*, 22:111–118, 1998.

[12] K. L. Clarkson. A modification of the greedy algorithm for vertex cover. *Information Processing Letters*, 16(1): 23–25, 1983.

[13] E. G. Coffman, M. R. Garey, D. S. Johnson, and A. S. LaPaugh. Scheduling file transfers. *SIAM Journal on Computing*, 14(3):744–780, 1985.

[14] W. J. Cook, W. H. Cunningham, W. R. Pullyblank, and A. Schrijver. *Combinatorial Optimization*. Wiley, 1998.

[15] R. Gandhi and J. Mestre. Combinatorial algorithms for data migration to minimize average completion time. In *Proceedings of the 9th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, pages 128–139, 2006.

[16] R. Gandhi, M. M. Halldórsson, G. Kortsarz, and H. Shachnai. Improved bounds for scheduling conflicting jobs with minsum criteria. In *Proceedings of the 2nd Workshop on Approximation and Online Algorithms (WAOA)*, pages 68–82, 2004.

[17] R. Gandhi, S. Khuller, and A. Srinivasan. Approximation algorithms for partial covering problems. *Journal of Algorithms*, 53(1):55–84, 2004.

[18] R. Gandhi, M. M. Halldórsson, G. Kortsarz, and H. Shachnai. Improved results for data migration and openshop scheduling. *ACM Transactions on Algorithms*, 2(1):116–129, 2006.

[19] M. X. Goemans and D. P. Williamson. *Approximation Algorithms for NP-Hard Problems*, chapter The primal-dual method for approximation algorithms and its application to network design problems, pages 144–191. PWS Publishing Company, 1997.

[20] J. Hall, J. Hartline, A. Karlin, J. Saia, and J. Wilkes. On algorithms for efficient data migration. In *Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 620–629, 2001.

[21] N. Immorlica, M. Mahdian, and V. S. Mirrokni. Cycle cover with short cycles. In *Proceedings of the 22nd Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 641–653, 2005.

[22] K. Jain, M. Mahdian, E. Markakis, A. Saberi, and V. V. Vazirani. Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP. *Journal of the ACM*, 50(6):795–824, 2003.

[23] S. Khuller, Y.-A. Kim, and Y. C. Wan. Algorithms for data migration with cloning. *SIAM Journal on Computing*, 33(2):448–461, 2004.

[24] Y.-A. Kim. Data migration to minimize the average completion time. *Journal of Algorithms*, 55:42–57, 2005.

[25] Y.-A. Kim, S. Khuller, and A. Malekian. Improved algorithms for data migration. In *Proceedings of the 9th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, pages 164–175, 2006.

[26] D. Marx. Complexity results for minimum sum edge coloring. Unpublished Manuscript, 2004.

[27] T. Nishizeki and K. Kashiwagi. On the 1.1 edge-coloring of multigraphs. *SIAM Journal on Discrete Mathematics*, 3(3):391–410, 1990.

[28] M. Queyranne and M. Sviridenko. New and improved algorithms for minsum shop scheduling. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 871–878, 2000.

[29] D. P. Williamson. The primal dual method for approximation algorithms. *Mathematical Programming*, 91(3): 447–478, 2002.

[30] D. P. Williamson, L. A. Hall, J. A. Hoogeveen, C. A. J. Hurkens, J. K. Lenstra, S. V. Sevast'janov, and D. B. Shmoys. Short shop schedules. *Operations Research*, 45 (2):288–94, 1997.

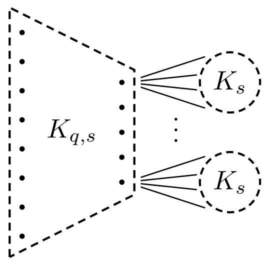


Figure 5: A bad instance for GREEDY-DEGREE.

A Bad example for a reasonable heuristic

In this section we study a seemingly reasonable heuristic for unweighted instances, which is based on the scheduling stage of ALR using the degrees instead of ℓ values to sort the edges. GREEDY-DEGREE first sorts the edges $(u, v) \in E$ in non-decreasing value of $\min(d_u, d_v)$, breaking ties with $\max(d_u, d_v)$; the edges are then scheduled (in sorted order) as early as possible without creating a conflict with the partial schedule built so far.

Gandhi and Mestre [15] showed a famlily of graphs for which an arbitrary greedy schedule can be a factor $\Omega(\sqrt[3]{n})$ away from optimum, where n is the number of vertices in the transfer graph. However, for those instances the above heuristic performs well—in fact it computes an optimal solution. Unfortunately GREEDY-DEGREE is not a constant factor approximation either.

LEMMA A.1. GREEDY-DEGREE *can produce a schedule with cost* $\Omega(\sqrt[4]{n})$ *times optimum.*

Proof. Our bad instance, shown in Figure 5, has three layers. The first, second and third layers contain q, s and s^2 nodes respectively. The first and second layer are connected with a complete bipartite graph $K_{q,s}$. The nodes in the third layer are divided into s groups, each forming a K_s that is connected to a single node in the second layer. The parameters q and s will be chosen to get the desired gap.

Notice that nodes in the first and third layers have degree s, thus, the heuristic first schedules the edges in the K_s's, and then we are free to schedule the remaining edges in *any* order as their endpoints have degree s and $q + s$. Suppose a solution S_1 first schedules the edges from the first to the second layer, while S_2 first schedules the edges from the second to the third layer. In S_1 the second-layer nodes are busy for the first q time steps working on the $K_{q,s}$ edges, as a result, every node in the third layer finishes by $\Omega(q)$, and the overall cost is $\Omega(s^2 q)$. On the other hand, in S_2 the third-layer nodes finish by $O(s)$ and the first and second-layer nodes finish by $O(q + s)$; thus, the overall cost is $O((q + s)^2 + s^3)$. Choosing $q = s^{\frac{3}{2}}$, the ratio of the cost of S_1 and S_2 is $\Omega(\sqrt{s})$. Our example has $O(s^2)$ nodes, therefore the greedy schedule can be an $\Omega(\sqrt[4]{n})$ factor away from optimum. $\square$