Improved Approximation Algorithms for Bipartite Correlation Clustering

Nir Ailon*[1], Noa Avigdor-Elgrabli[1], Edo Liberty[2], and Anke van Zuylen[3]

[1] Technion, Haifa, Israel [nailon|noaelg]@cs.technion.ac.il
[2] Yahoo! Research, Haifa, Israel edo.liberty@ymail.com
[3] Max-Planck Institut für Informatik, Saarbrücken, Germany anke@mpi-inf.mpg.de

Abstract. In this work we study the problem of Bipartite Correlation Clustering (BCC), a natural bipartite counterpart of the well studied Correlation Clustering (CC) problem. Given a bipartite graph, the objective of BCC is to generate a set of vertex-disjoint bi-cliques (clusters) which minimizes the symmetric difference to it. The best known approximation algorithm for BCC due to Amit (2004) guarantees an 11-approximation ratio.[4]

In this paper we present two algorithms. The first is an improved 4-approximation algorithm. However, like the previous approximation algorithm, it requires solving a large convex problem which becomes prohibitive even for modestly sized tasks.

The second algorithm, and our main contribution, is a simple randomized combinatorial algorithm. It also achieves an expected 4-approximation factor, it is trivial to implement and highly scalable. The analysis extends a method developed by Ailon, Charikar and Newman in 2008, where a randomized pivoting algorithm was analyzed for obtaining a 3-approximation algorithm for CC. For analyzing our algorithm for BCC, considerably more sophisticated arguments are required in order to take advantage of the bipartite structure.

Whether it is possible to achieve (or beat) the 4-approximation factor using a scalable *and* deterministic algorithm remains an open problem.

1 Introduction

The analysis of large bipartite graphs is becoming of increased practical importance. Recommendation systems, for example, take as input a large dataset of bipartite relations between users and objects (e.g. movies, goods) and analyze its structure for the purpose of predicting future relations [2]. Other examples may include images vs. user generated tags and search engine queries vs. search results. Bipartite clustering is also studied in the context of gene expression data analysis (see e.g. [3][4][5] and references therein). In spite of the extreme practical

* Supported in part by the Yahoo! Faculty Research and Engagement Program

[4] A previously claimed 4-approximation algorithm [1] is erroneous, as we show in Appendix A.

importance of bipartite clustering, far less is known about it than standard (non-bipartite) clustering. Many notions of clustering bipartite data exist. Some aim at finding the best cluster, according to some definition of 'best'. Others require that the entire data (graph) be represented as clusters. Moreover, data points (nodes) may either be required to belong to only one cluster or allowed to belong to different overlapping clusters. Here the goal is to obtain non-overlapping (vertex disjoint) clusters covering the entire input vertex set. Hence, one may think of our problem as *bipartite graph partitioning*.

In Bipartite Correlation Clustering (BCC) we are given a bipartite graph as input, and output a set of disjoint clusters covering the graph nodes. Clusters may contain nodes from either side of the graph, but they may possibly contain nodes from only one side. We think of a cluster as a bi-clique connecting all the elements from its left and right counterparts. An output clustering is hence a union of bi-cliques covering the input node set. The cost of the solution is the symmetric difference between the input and output edge sets. Equivalently, any pair of vertices, one on the left and one of the right, will incur a unit cost if either (1) they are connected by an input edge but the output clustering separates them into distinct clusters, or (2) they are not connected by an input edge but the output clustering assigns them to the same cluster. The objective is to minimize this cost. This problem formulation is the bipartite counterpart of the more well known Correlation Clustering (CC), introduced by Bansal, Blum and Chawla [6], where the objective is to cover the node set of a (non-bipartite) graph with disjoint cliques (clusters) minimizing the symmetric difference with the given edge set. One advantage of this objective is in alleviating the need to specify the number of output clusters, as often needed in clustering settings such as k-means or k-median. Another advantage lies in the objective function, which naturally corresponds to some models about noise in the data. Examples of applications include [7], where a reduction from *consensus clustering* to our problem is introduced, and [8] for an application of a related problem to large scale document-term relation analysis.

Bansal et. al [6] gave a $c \approx 10^4$ factor for approximating CC running in time $O(n^2)$ where n is the number of nodes in the graph. Later, Demaine, Emanuel, Fiat and Immorlica [9] gave a $O(\log(n))$ approximation algorithm for an *incomplete* version of CC, relying on solving an LP and rounding its solution by employing a region growing procedure. By incomplete we mean that only a subset of the node pairs participate in the symmetric difference cost calculation.[5] BCC is, in fact, a special case of incomplete CC, in which the non-participating node pairs lie on the same side of the graph. Charikar, Guruswami and Wirth [10] provide a 4-approximation algorithm for CC, and another $O(\log n)$-approximation algorithm for the incomplete case. Later, Ailon, Charikar and Newman [11] provided a 2.5-approximation algorithm for CC based on rounding an LP. They also provide a simpler 3-approximation algorithm, QuickCluster, which runs in

[5] In some of the literature, CC refers to the much harder incomplete version, and "CC in complete graphs" is used for the version we have described here.

time linear in the number of edges of the graph. In [12] it was argued that QuickCluster runs in expected time $O(n + cost(OPT))$.

Van Zuylen and Williamson [13] provided de-randomization for the algorithms presented in [11] with no compromise in the approximation guarantees. Giotis and Guruswami [14] gave a PTAS for the CC case in which the number of clusters is constant. Later, (using other techniques) Karpinski and Schudy [15] improved the runtime.

Amit [3] was the first to address BCC directly. She proved its NP-hardness and gave a constant 11-approximation algorithm based on rounding a linear programming in the spirit of Charikar et. al's [10] algorithm for CC.

It is worth noting that in [1] a 4-approximation algorithm for BCC was presented and analyzed. The presented algorithm is incorrect (we give a counter example in the paper) but their attempt to use arguments from [11] is an excellent one. We will show how to achieve the claimed guarantee with an extension of the method in [11].

1.1 Our Results

We first describe a deterministic 4-approximation algorithm for BCC (Section 2). It starts by solving a Linear Program in order to convert the problem to a non bipartite instance (CC) and then uses the pivoting algorithm [13] to construct a clustering. The algorithm is similar to the one in [11] where nodes from the graph are chosen randomly as 'pivots' or 'centers' and clusters are generated from their neighbor sets. Arguments from [13] derandomize this choice and give us a deterministic 4-approximation algorithm. This algorithm, unfortunately, becomes impractical for large graphs. The LP solved in the first step needs to enforce the transitivity of the clustering property for all sets of three nodes and thus contains $\Omega(n^3)$ constraints.

Our main contribution is an extremely simple combinatorial algorithm called PivotBiCluster which achieves the same approximation guarantee. The algorithm is straightforward to implement and terminates in $O(|E|)$ operations (the number of edges in the graph). We omit the simple proof of the running time since it is immediate given the algorithm's description, see Section 3.2. A disadvantage of PivotBiCluster is the fact that it is randomized and achieves the approximation guarantee only in expectation. However, a standard Markov inequality argument shows that taking the best solution obtained from independent repetitions of the algorithm achieves an approximation guarantee of $4 + \varepsilon$ for any constant $\varepsilon > 0$.

While the algorithm itself is simple, its proof is rather involved and requires a significant extension of previously developed techniques. To explain the main intuition behind our approach, we recall the method of Ailon et. al [11]. The algorithm for CC presented there (the unweighted case) is as follows: choose a random vertex, form a cluster including it and its neighbors, remove the cluster from the graph, and repeat until the graph is empty. This random-greedy algorithm returns a solution with cost at most 3 times that of the optimal solution, in expectation. The key to the analysis is the observation that each part of the

cost of the algorithm's solution can be naturally related to a certain *minimal contradicting structure*, for CC, an induced subgraph of 3 vertices and exactly 2 edges. Notice that in any such structure, at least one vertex pair must be violated. A vertex pair being violated means it contributes to the symmetric difference between the graph and the clustering. In other words, the vertex pairs that a clustering violates must *hit* the set of minimal contradicting structures. A corresponding hitting set LP lower bounding the optimal solution was defined to capture this simple observation. The analysis of the random-greedy solution constructs a dual feasible solution to this LP, using probabilities arising in the algorithm's probability space.

It is tempting here to consider the corresponding minimal contradicting structure for BCC, namely a set of 4 vertices, 2 on each side, with exactly 3 edges between them. Unfortunately, this idea turned out to be evasive. A proposed solution attempting this [1] has a counter example which we describe and analyze in Appendix A and is hence incorrect. Our attempts to follow this path have also failed. In our analysis we resorted to contradicting structures of unbounded size. Such a structure consists of two vertices ℓ_1, ℓ_2 of the left side and two sets of vertices N_1, N_2 on the right hand side such that N_i is contained in the neighborhood of ℓ_i for $i = 1, 2$, $N_1 \cap N_2 \neq \emptyset$ and $N_1 \neq N_2$. We define a hitting LP as we did earlier, this time of possibly exponential size, and analyze its dual in tandem with a carefully constructed random-greedy algorithm, PivotBiCluster. At each round PivotBiCluster chooses a random pivot vertex on the left, constructs a cluster with its right hand side neighbors, and for each other vertex on the left randomly decides whether to join the new cluster or not. The new cluster is removed and the process is repeated until the graph is exhausted. The main challenge is to find joining probabilities of left nodes to new clusters which can be matched to a feasible solution to the dual LP.

1.2 Paper Structure

We first present a deterministic LP rounding based algorithm in Section 2. Our main algorithm in given in Section 3. We start with notations and definitions in Section 3.1, followed by the algorithm's description and our main theorem in Section 3.2. The algorithm's analysis is logically partitioned between Sections 3.3, 3.4, and 3.5. Finally, we propose future research and conjectures in Section 4.

2 A deterministic LP rounding algorithm

We start with a deterministic algorithm with a 4-approximation guarantee by directly rounding an optimal solution to a linear programming relaxation LP_{det} of BCC. Let the input graph be $G = (L, R, E)$ where L and R are the sets of left and right nodes and E be a subset of $L \times R$. For notational purposes, we define the following constants given our input graph: for each edge $(i, j) \in E$ we define $w_{ij}^+ = 1, w_{ij}^- = 0$ and for each non-edge $(i, j) \notin E$ we define $w_{ij}^+ = 0, w_{ij}^- = 1$. Our integer program has an indicator variable y_{ij}^+ which equals 1 iff i and j are placed

in the same cluster. The variable is defined for each pair of vertices, *and not only* for pairs (ℓ, r) with $\ell \in L, r \in R$. Hence, in a certain sense, this approach forgets about bipartiteness. For ease of notation we define $y^-_{ij} = 1 - y^+_{ij}$. The objective function becomes $\sum_{(i,j)}(w^+_{ij}y^-_{ij} + w^-_{ij}y^+_{ij})$. The clustering consistency constraint is given as $y^-_{ij} + y^-_{jk} + y^+_{ik} \geq 1$ for all (ordered) sets of three vertices $i, j, k \in V$, where $V = L \cup R$. The relaxed LP is given by:

$$\begin{aligned} \mathrm{LP}_{\mathrm{det}} = \min & \sum_{(i,j)} (w^+_{ij}y^-_{ij} + w^-_{ij}y^+_{ij}) \\ \text{s.t} \ \ & \forall i,j,k \in V : y^-_{ij} + y^-_{jk} + y^+_{ik} \geq 1, y^+_{ij} + y^-_{ij} = 1, y^+_{ij}, y^-_{ij} \in [0,1]. \end{aligned}$$

Given an optimal solution to $\mathrm{LP}_{\mathrm{det}}$, we partition the pairs of distinct vertices into two sets E^+ and E^-, where $e \in E^+$ if $y^+_e \geq \frac{1}{2}$ and $e \in E^-$ otherwise. Since each distinct pair is in either E^+ or E^-, we have an instance of CC which can then be clustered using the algorithm of Van Zuylen and Williamson [13]. The algorithm is a derandomization of Ailon et al's [11] randomized QuickCluster for CC. QuickCluster recursively constructs a clustering simply by iteratively choosing a pivot vertex i at random, forming a cluster C that contains i and all vertices j such that $(i, j) \in E^+$, removing them from the graph and repeating. Van Zuylen and Williamson [13] replace the random choice of pivot by a deterministic one, and show conditions under which the resulting algorithm output is a constant factor approximation with respect to the LP objective function. To describe their choice of pivot, we need the notion of a "bad triplet" [11]. We will call a triplet (i, j, k) a bad triplet if exactly two of the pairs among $\{(i, j), (j, k), (k, i)\}$ are edges in E^+. Consider the pairs of vertices on which the output of QuickCluster disagrees with E^+ and E^-, i.e., pairs $(i, j) \in E^-$ that are in the same cluster, and pairs $(i, j) \in E^+$ that are not in the same cluster in the output clustering. It is not hard to see that in both cases, there was some call to QuickCluster in which (i, j, k) formed a bad triplet with the pivot vertex k. The pivot chosen by Van Zuylen and Williamson [13] is the pivot that minimizes the ratio of the weight of the edges that are in a bad triplet with the pivot and their LP contribution.

Given an optimal solution y to $\mathrm{LP}_{\mathrm{det}}$ we let $c_{ij} = w^+_{ij}y^-_{ij} + w^-_{ij}y^+_{ij}$. Recall that E^+, E^- are also defined by the optimal solution. We are now ready to present the deterministic LP rounding algorithm:

Theorem 1. *[From Van Zuylen et al. [13]] Algorithm QuickCluster* (V, E^+, E^-) *from [11] returns a solution with cost at most* 4 *times the cost of the optimal solution to* $\mathrm{LP}_{\mathrm{det}}$ *if in each iteration a pivot vertex is chosen that minimizes :* $F(k) = \left(\sum_{(i,j)\in E^+:(i,j,k)\in\mathcal{B}} w^+_{ij} + \sum_{(i,j)\in E^-:(i,j,k)\in\mathcal{B}} w^-_{ij}\right) / \sum_{(i,j,k)\in\mathcal{B}} c_{ij}$, *where* $\mathcal{B}$ *is the set of bad triplets on vertices that haven't been removed from the graph in previous steps.*

The proof of Theorem 1 is deferred to Appendix B.

3 The Combinatorial 4-Approximation Algorithm

3.1 Notation

Before describing the framework we give some general facts and notations. Let the input graph again be $G = (L, R, E)$ where L and R are the sets of left and right nodes and E be a subset of $L \times R$. Each element $(\ell, r) \in L \times R$ will be referred to as a *pair*.

A solution to our combinatorial problem is a clustering $C_1, C_2, \ldots, C_m$ of the set $L \cup R$. We identify such a clustering with a bipartite graph $B = (L, R, E_B)$ for which $(\ell, r) \in E_B$ if and only if $\ell \in L$ and $r \in R$ are in the same cluster C_i for some i. Note that given B, we are unable to identify clusters contained exclusively in L (or R), but this will not affect the cost. We therefore take the harmless decision that nodes in single-side clusters are always singletons.

We will say that a pair $e = (\ell, r)$ is violated if $e \in (E \setminus E_B) \cup (E_B \setminus E)$. For convenience, let $x_{G,B}$ be the indicator function for the violated pair set, i.e., $x_{G,B}(e) = 1$ if e is violated and 0 otherwise. We will also simply use $x(e)$ when it is obvious to which graph G and clustering B it refers. The cost of a clustering solution is defined to be $\text{cost}_G(B) = \sum_{e \in L \times R} x_{G,B}(e)$. Similarly, we will use $\text{cost}(B) = \sum_{e \in L \times R} x(e)$ when G is clear from the context, Let $N(\ell) = \{r | (\ell, r) \in E\}$ be the set of all right nodes adjacent to ℓ.

It will be convenient for what follows to define a *tuple*. We define a tuple T to be $(\ell_1^T, \ell_2^T, R_1^T, R_{1,2}^T, R_2^T)$ where $\ell_1^T, \ell_2^T \in L$, $\ell_1^T \neq \ell_2^T$, $R_1^T \subseteq N(\ell_1^T) \setminus N(\ell_2^T)$, $R_2^T \subseteq N(\ell_2^T) \setminus N(\ell_1^T)$ and $R_{1,2}^T \subseteq N(\ell_2^T) \cap N(\ell_1^T)$. In what follows, we may omit the superscript of T. Given a tuple $T = (\ell_1^T, \ell_2^T, R_1^T, R_{1,2}^T, R_2^T)$, we define the *conjugate tuple* $\bar{T} = (\ell_1^{\bar{T}}, \ell_2^{\bar{T}}, R_1^{\bar{T}}, R_{1,2}^{\bar{T}}, R_2^{\bar{T}}) = (\ell_2^T, \ell_1^T, R_2^T, R_{1,2}^T, R_1^T)$. Note that $\bar{\bar{T}} = T$.

3.2 Algorithm Description

We now describe PivotBiCluster. The algorithm runs in rounds. In every round it creates one cluster and possibly many singletons, all of which are removed from the graph before continuing to the next iteration. Abusing notation, by $N(\ell)$ we mean, in the algorithm description, all the neighbors of $\ell \in L$ which have not yet been removed from the graph.

Every such cycle performs two phases. In the first phase, PivotBiCluster picks a node on the left side uniformly at random, ℓ_1, and forms a new cluster $C = \{\ell_1\} \cup N(\ell_1)$. This will be referred to as the ℓ_1-phase and ℓ_1 will be referred to as the left center of the cluster. In the second phase, denoted as the ℓ_2-sub-phase corresponding to the ℓ_1-phase, the algorithm iterates over all other remaining left nodes, ℓ_2, and decides either to (1) append them to C, (2) turn them into singletons, or (3) do nothing. We now explain how to make this decision. let $R_1 = N(\ell_1) \setminus N(\ell_2)$, $R_2 = N(\ell_2) \setminus N(\ell_1)$ and $R_{1,2} = N(\ell_1) \cap N(\ell_2)$. With probability $\min\{\frac{|R_{1,2}|}{|R_2|}, 1\}$ do one of two things: (1) If $|R_{1,2}| \geq |R_1|$ append ℓ_2 to C, and otherwise (2) (if $|R_{1,2}| < |R_1|$), turn ℓ_2 into a singleton. In the remaining

probability, (3) do nothing for ℓ_2, leaving it in the graph for future iterations. Examples for cases the algorithm encounters for different ratios of R_1, $R_{1,2}$, and R_2 are given in Figure 3.2.

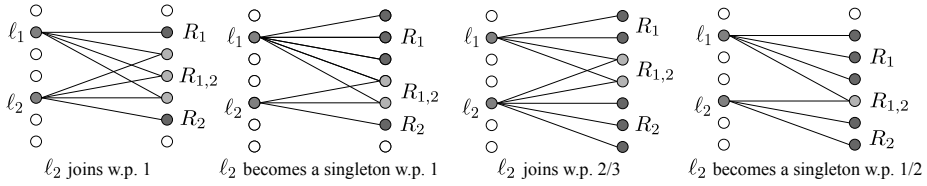

Fig. 1. Four example cases in which ℓ_2 either joins the cluster created by ℓ_1 or becomes a singleton. In the two right most examples, with the remaining probability nothing is decided about ℓ_2.

Theorem 2. *Algorithm PivotBiCluster returns a solution with expected cost at most* 4 *times that of the optimal solution.*

3.3 Algorithm Analysis

We start by describing *bad events*. This will help us relate the expected cost of the algorithm to a sum of event probabilities and expected consequent costs.

Definition 1. *We say that a* bad event*, X_T, happens to the tuple $T = (\ell_1^T, \ell_2^T, R_1^T, R_{1,2}^T, R_2^T)$ if during the execution of PivotBiCluster, ℓ_1^T was chosen to be a left center while ℓ_2^T was still in the graph, and* at that moment*, $R_1^T = N(\ell_1^T) \setminus N(\ell_2^T)$, $R_{1,2}^T = N(\ell_1^T) \cap N(\ell_2^T)$, and $R_2^T = N(\ell_2^T) \setminus N(\ell_1^T)$. (We refer by $N(\cdot)$ here to the neighborhood function in a particular moment of the algorithm execution.)*

If a bad event X_T happens to tuple T, we *color* the following pairs with color T: (1) $\{(\ell_2^T, r) \ : \ r \in R_1^T \cup R_{1,2}^T\}$, (2) $\{(\ell_2^T, r) \ : \ r \in R_2^T\}$. We color the latter pairs only if we decide to associate ℓ_2^T to ℓ_1^T's cluster, or if we decide to make ℓ_2^T a singleton during the ℓ_2-sub-phase corresponding to the ℓ_1-phase. Notice that these pairs are the remaining pairs (in the beginning of event X_T) from ℓ_2^T that after the ℓ_2^T-sub-phase will be removed from the graph. We also denote by $X_{e,T}$ the event that the edge e is colored with color T.

Lemma 1. *During the execution of PivotBiCluster each pair $(\ell, r) \in L \times R$ is colored at most once, and each violated pair is colored exactly once.*

Proof. For the first part, we show that pairs are colored at most once. A pair (ℓ, r) can only be colored during an ℓ_2-sub-phases with respect to some ℓ_1-phase,

if $\ell = \ell_2$. Clearly, this will only happen in one ℓ_1-phase, as every time a pair is labeled either ℓ_2 or r are removed from the graph. Indeed, either $r \in R_1 \cup R_{1,2}$ in which case r is removed, or $r \in R_2$, but then ℓ is removed since it either joins the cluster created by ℓ_1 or becomes a singleton. For the second part, note that during each ℓ_1-phase the only pairs removed from the graph not colored are between left centers, ℓ_1, and right nodes in the graph at that time. All of these pairs are clearly not violated. □

We denote by q_T the probability that event X_T occurs and by $\text{cost}(T)$ the number of violated pairs that are colored by X_T. From Lemma 1, we get:

Corollary 1. *Letting random variable* COST *denote* $\text{cost}(PivotBiCluster)$*:*

$$\mathbb{E}[\text{COST}] = \mathbb{E}\left[\sum_{e \in L \times R} x(e)\right] = \mathbb{E}\left[\sum_{T} \text{cost}(T)\right] = \sum_{T} q_T \cdot \mathbb{E}[\text{cost}(T)|X_T] \ .$$

3.4 Contradicting Structures

We now identify bad structures in the graph for which every output must incur some cost, and use them to construct an LP relaxation for our problem. In the case of BCC the minimal such structures are "bad squares": A set of four nodes, two on each side, between which there are only three edges. We make the trivial observation that any clustering B must make at least one mistake on any such bad square, s (we think of s as the set of 4 pairs connecting its two left nodes and two right nodes). Any clustering solution's violating pair set must hit these squares. Let S denote the set of all bad squares in the input graph G.

It will not be enough for our purposes to concentrate on squares in our analysis. Indeed, at an ℓ_2-sub-phase, decisions are made based on the intersection pattern of the current neighborhoods of ℓ_2 and ℓ_1 - a possibly unbounded structure. The *tuples* now come in handy.

Consider tuple $T = (\ell_1^T, \ell_2^T, R_1^T, R_{1,2}^T, R_2^T)$ for which $|R_{1,2}^T| > 0$ and $|R_2^T| > 0$. Notice that for every selection of $r_2 \in R_2^T$, and $r_{1,2} \in R_{1,2}^T$ the tuple contains the bad square induced by $\{\ell_1, r_2, \ell_2, r_{1,2}\}$. Note that there may also be bad squares $\{\ell_2, r_1, \ell_1, r_{1,2}\}$ for every $r_1 \in R_1^T$ and $r_{1,2} \in R_{1,2}^T$ but these will be associated to the *conjugate tuple* $\bar{T} = (\ell_2^T, \ell_1^T, R_2^T, R_{1,2}^T, R_1^T)$.

For each tuple we can write a corresponding linear constraint for the vector $\{x(e) : e \in L \times R\}$, indicating, as we explained above, the pairs the algorithm violates. A tuple constraint is the sum of the constraints of all bad squares it is associated with, where a constraint for square s is simply defined as $\sum_{e \in s} x(e) \geq 1$. The purpose of this constraint is to encode that we must violate at least one pair in a bad square. Since each tuple corresponds to $|R_2^T| \cdot |R_{1,2}^T|$ bad squares, we get the following constraint:

$$\forall\ T: \quad \sum_{r_2 \in R_2^T, r_{1,2} \in R_{1,2}^T} \left(x_{\ell_1^T, r_2} + x_{\ell_1^T, r_{1,2}} + x_{\ell_2^T, r_2} + x_{\ell_2^T, r_{1,2}}\right) =$$

$$\sum_{r_2 \in R_2^T} |R_{1,2}^T| \cdot (x_{\ell_1^T, r_2} + x_{\ell_2^T, r_2}) + \sum_{r_{1,2} \in R_{1,2}^T} |R_2^T| \cdot (x_{\ell_1^T, r_{1,2}} + x_{\ell_2^T, r_{1,2}}) \geq |R_2^T| \cdot |R_{1,2}^T|$$

The following linear program hence provides a lower bound for the optimal solution: $LP = \min \sum_{e \in L \times R} x(e)$

$$\text{s.t. } \forall T \ \frac{1}{|R_2^T|} \sum_{r_2 \in R_2^T} (x_{\ell_1^T, r_2} + x_{\ell_2^T, r_2}) + \frac{1}{|R_{1,2}^T|} \sum_{r_{1,2} \in R_{1,2}^T} (x_{\ell_1^T, r_{1,2}} + x_{\ell_2^T, r_{1,2}}) \geq 1$$

The dual program is as follows: $DP = max \sum_T \beta(T)$

$$\text{s.t. } \forall (\ell, r) \in E :$$
$$\sum_{T:\, \ell_2^T = \ell, r \in R_2^T} \frac{\beta(T)}{|R_2^T|} + \sum_{T:\, \ell_1^T = \ell,\, r \in R_{1,2}^T} \frac{\beta(T)}{|R_{1,2}^T|} + \sum_{T:\, \ell_2^T = \ell,\, r \in R_{1,2}^T} \frac{\beta(T)}{|R_{1,2}^T|} \leq 1 \quad (1)$$
$$\text{and } \forall (\ell, r) \notin E : \sum_{T:\, \ell_1^T = \ell,\, r \in R_2^T} \frac{1}{|R_2^T|} \beta(T) \leq 1 \quad (2)$$

3.5 Obtaining the Competitive Analysis

We now relate the expected cost of the algorithm on each tuple to a feasible solution to DP. We remind the reader that q_T denotes the probability of event X_T corresponding to tuple T.

Lemma 2. *The solution* $\beta(T) = \alpha_T \cdot q_T \cdot \min\{|R_{1,2}^T|, |R_2^T|\}$ *is a feasible solution to* DP, *when* $\alpha_T = \min\left\{1, \frac{|R_{1,2}^T|}{\min\{|R_{1,2}^T|, |R_1^T|\} + \min\{|R_{1,2}^T|, |R_2^T|\}}\right\}$.

Proof. First, notice that given a pair $e = (\ell, r) \in E$ each tuple T can appear in at most one of the sums in the LHS of the DP constraints (1) (as $R_1^T, R_{1,2}^T, R_2^T$ are disjoint). We distinguish between two cases.

1. Consider T appearing in the first sum of the LHS of (1), meaning that $\ell_2^T = \ell$ and $r \in R_2^T$. e is colored with color T if ℓ_2^T joined the cluster of ℓ_1^T or if ℓ_2 was turned into a singleton. Both cases happen, conditioned on X_T, with probability $\Pr[X_{e,T} | X_T] = \min\left\{\frac{|R_{1,2}^T|}{|R_2^T|}, 1\right\}$. Thus, we can bound the contribution of T to the sum as follows:
$$\begin{aligned}\frac{1}{|R_2^T|}\beta(T) &= \frac{1}{|R_2^T|}\alpha_T \cdot q_T \cdot \min\{|R_{1,2}^T|, |R_2^T|\} \leq q_T \cdot \min\left\{\frac{|R_{1,2}^T|}{|R_2^T|}, 1\right\} \\ &= \Pr[X_T] \Pr[X_{e,T} | X_T] = \Pr[X_{e,T}].\end{aligned}$$
(The inequality is simply because $\alpha_T \leq 1$.)
2. T contributes to the second or third sum in the LHS of (1). By definition of the conjugate $\bar{T}$, the following holds:
$$\sum_{T \ s.t \ \ell_1^T = \ell,\, r \in R_{1,2}^T} \frac{\beta(T)}{|R_{1,2}^T|} + \sum_{T \ s.t \ \ell_2^T = \ell,\, r \in R_{1,2}^T} \frac{\beta(T)}{|R_{1,2}^T|} = \sum_{T \ s.t \ \ell_1^T = \ell,\, r \in R_{1,2}^T} \frac{\left(\beta(T) + \beta(\bar{T})\right)}{|R_{1,2}^T|}.$$

It is therefore sufficient to bound the contribution of each T to the RHS of the latter equality. We henceforth focus on tuples T for which $\ell = \ell_1^T$ and $r \in R_{1,2}^T$. Consider a moment in the algorithm execution in which both ℓ_1^T and ℓ_2^T were still present in the graph, $R_1^T = N(\ell_1^T) \setminus N(\ell_2^T)$, $R_{1,2}^T = N(\ell_1^T) \cap N(\ell_2^T)$, $R_2^T = N(\ell_2^T) \setminus N(\ell_1^T)$ and one of ℓ_1^T, ℓ_2^T was chosen to be a left center.[6] Either one of ℓ_1^T and ℓ_2^T had the same probability to be chosen. In other words, $\Pr[X_T | X_T \cup X_{\bar{T}}] = \Pr[X_{\bar{T}} | X_T \cup X_{\bar{T}}]$, and hence, $q_T = q_{\bar{T}}$. Further, notice that $e = (\ell, r)$ is never colored with color T, and if event $X_{\bar{T}}$ happens then e is colored with color $\bar{T}$ with probability 1. Therefore:

$$\frac{1}{|R_{1,2}^T|}\left(\beta(T) + \beta(\bar{T})\right) = \frac{1}{|R_{1,2}^T|} \cdot q_T \cdot \min\left\{1, \frac{|R_{1,2}^T|}{\min\{|R_{1,2}^T|, |R_1^T|\} + \min\{|R_{1,2}^T|, |R_2^T|\}}\right\} \cdot \left(\min\{|R_{1,2}^T|, |R_2^T|\} + \min\{|R_{1,2}^{\bar{T}}|, |R_2^{\bar{T}}|\}\right)$$
$$\leq q_T = q_{\bar{T}} = \Pr[X_{\bar{T}}] = \Pr[X_{e,\bar{T}}] + \Pr[X_{e,T}].$$

Summing this all together, for every edge $e \in E$:

$$\sum_{T \ s.t \ \ell_2^T = \ell, r \in R_2^T} \frac{\beta(T)}{|R_2^T|} + \sum_{T \ s.t \ \ell_1^T = \ell, \, r \in R_{1,2}^T} \frac{\beta(T)}{|R_{1,2}^T|} + \sum_{T \ s.t \ \ell_2^T = \ell, \, r \in R_{1,2}^T} \frac{\beta(T)}{|R_{1,2}^T|} \leq \sum_T Pr[X_{e,T}].$$

By the first part of Lemma 1 we know that $\sum_T Pr[X_{e,T}]$ is exactly the probability of the edge e to be colored (the sum is over probabilities of disjoint events), therefore it is at most 1, as required to satisfy (1).

Now consider a pair $e = (\ell, r) \notin E$. A tuple T contributes to (2) if $\ell_1^T = \ell$ and $r \in R_2^T$. Since, as before, $q_T = q_{\bar{T}}$ and since $\Pr[X_{e,\bar{T}} | X_{\bar{T}}] = 1$ (this follows from the first coloring rule described in the beginning of Section 3.3) we obtain the following:

$$\sum_{T \ s.t \ \ell_1^T = \ell, \, r \in R_2^T} \frac{1}{|R_2^T|}\beta(T) = \sum_{T \ s.t \ \ell_1^T = \ell, \, r \in R_2^T} \frac{1}{|R_2^T|} \cdot \alpha_T \cdot q_T \cdot \min\{|R_{1,2}^T|, |R_2^T|\}$$
$$\leq \sum_{T \ s.t \ \ell_1^T = \ell, \, r \in R_2^T} q_T \ = \sum_{\bar{T} \ s.t \ \ell_2^{\bar{T}} = \ell, \, r \in R_1^{\bar{T}}} q_{\bar{T}} = \sum_{\bar{T} \ s.t \ \ell_2^{\bar{T}} = \ell, \, r \in R_1^{\bar{T}}} \Pr[X_{\bar{T}}]$$
$$= \sum_{\bar{T} \ s.t \ \ell_2^{\bar{T}} = \ell, \, r \in R_1^{\bar{T}}} \Pr[X_{e,\bar{T}}] = \sum_T \Pr[X_{e,T}].$$

From the same reason as before, this is at most 1, as required for (2). □

After presenting the feasible solution to our dual program, we have left to prove that the expected cost of PivotBiCluster is at most 4 times the DP value of this solution. For this we need the following:

[6] Recall that $N(\cdot)$ depends on the "current" state of the graph at that moment, after removing previously created clusters.

Lemma 3. *For any tuple T,*
$q_T \cdot \mathbb{E}[\text{cost}(T)|X_T] + q_{\bar{T}} \cdot \mathbb{E}[\text{cost}(\bar{T})|X_{\bar{T}}] \leq 4 \cdot \left(\beta(T) + \beta(\bar{T})\right)$.

Proof. We consider three cases, according to the structure of T.
Case 1. $|R_1^T| \leq |R_{1,2}^T|$ and $|R_2^T| \leq |R_{1,2}^T|$ (equivalently $|R_1^{\bar{T}}| \leq |R_{1,2}^{\bar{T}}|$ and $|R_2^{\bar{T}}| \leq |R_{1,2}^{\bar{T}}|$): For this case, $\alpha_T = \alpha_{\bar{T}} = \min\left\{1, \frac{|R_{1,2}^T|}{|R_1^T|+|R_2^T|}\right\}$, and we get that

$$\begin{aligned}\beta(T) + \beta(\bar{T}) &= \alpha_T \cdot q_T \cdot \left(\min\{|R_{1,2}^T|, |R_2^T|\} + \min\{|R_{1,2}^T|, |R_1^T|\}\right) \\ &= q_T \cdot \min\{(|R_2^T| + |R_1^T|), |R_{1,2}^T|\} \geq \frac{1}{2} \cdot q_T \cdot (|R_2^T| + |R_1^T|).\end{aligned}$$

Since $|R_1^T| \leq |R_{1,2}^T|$, if event X_T happens PivotBiCluster adds ℓ_2^T to ℓ_1^T's cluster with probability $\min\left\{\frac{|R_{1,2}^T|}{|R_2^T|}, 1\right\} = 1$. Therefore the pairs colored with color T that PivotBiCluster violates are all the edges from ℓ_2^T to R_2^T and all the non-edges from ℓ_2^T to R_1^T, namely, $|R_2^T| + |R_1^T|$ edges. The same happens in the event $X_{\bar{T}}$ as the conditions on $|R_1^{\bar{T}}|$, $|R_{1,2}^{\bar{T}}|$, and $|R_2^{\bar{T}}|$ are the same, and since $|R_2^{\bar{T}}| + |R_1^{\bar{T}}| = |R_1^T| + |R_2^T|$. Thus,

$$q_T \cdot \left(\mathbb{E}[\text{cost}(T|X_T)] + \mathbb{E}[\text{cost}(\bar{T}|X_{\bar{T}})]\right) = q_T \left(2\left(|R_2^T| + |R_1^T|\right)\right) \leq 4 \cdot \left(\beta(T) + \beta(\bar{T})\right).$$

Case 2. $|R_1^T| < |R_{1,2}^T| < |R_2^T|$ (equivalently $|R_1^{\bar{T}}| > |R_{1,2}^{\bar{T}}| > |R_2^{\bar{T}}|$)[7]: We defer this case to Appendix C due to lack of space.
Case 3. $|R_{1,2}^T| < |R_1^T|$ and $|R_{1,2}^T| < |R_2^T|$ (equivalently, $|R_{1,2}^{\bar{T}}| < |R_2^{\bar{T}}|$ and $|R_{1,2}^{\bar{T}}| < |R_1^{\bar{T}}|$): We defer this case to Appendix D due to lack of space.

□

By Corollary 1: $\mathbb{E}[\text{cost}(PivotBiCluster)] = \sum_T \Pr[X_T] \cdot \mathbb{E}[\text{cost}(T)|X_T]$

$$= \frac{1}{2} \sum_T \left(\Pr[X_T] \cdot \mathbb{E}[\text{cost}(T)|X_T] + \Pr[X_{\bar{T}}] \cdot \mathbb{E}[\text{cost}(\bar{T})|X_{\bar{T}}]\right).$$

By Lemma 3 the above RHS is at most $2 \cdot \sum_T (\beta(T) + \beta(\bar{T})) = 4 \cdot \sum_T \beta(T)$. We conclude that $\mathbb{E}[\text{cost}(PivotBiCluster)] \leq 4 \cdot \sum_T \beta(T) \leq 4 \cdot OPT$. This proves our main Theorem 2.

4 Future Work

The main open problem is that of improving the factor 4 approximation ratio. We believe that it should be possible by using both symmetry and bipartiteness simultaneously. Indeed, our LP rounding algorithm in Section 2 is symmetric with respect to the left and right sides of the graph. However, in a sense, it "forgets" about bipartiteness altogether. On the other hand, our combinatorial algorithm in Section 3 uses bipartiteness in a very strong way but is asymmetric which is counterintuitive.

[7] From symmetry reasons (between T and $\bar{T}$) here we also deal with the case $|R_2^T| < |R_{1,2}^T| < |R_1^T|$.

References

1. Jiong Guo, Falk Hüffner, Christian Komusiewicz, and Yong Zhang. Improved algorithms for bicluster editing. In *TAMC'08: Proceedings of the 5th international conference on Theory and applications of models of computation*, pages 445–456, Berlin, Heidelberg, 2008. Springer-Verlag.
2. Panagiotis Symeonidis, Alexandros Nanopoulos, Apostolos Papadopoulos, and Yannis Manolopoulos. Nearest-biclusters collaborative filtering, 2006.
3. Noga Amit. The bicluster graph editing problem, 2004.
4. Sara C. Madeira and Arlindo L. Oliveira. Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 1:24–45, January 2004.
5. Yizong Cheng and George M. Church. Biclustering of expression data. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pages 93–103. AAAI Press, 2000.
6. Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *Machine Learning*, 56:89–113, 2004.
7. Xiaoli Zhang Fern and Carla E. Brodley. Solving cluster ensemble problems by bipartite graph partitioning. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, page 36, New York, NY, USA, 2004. ACM.
8. Hongyuan Zha, Xiaofeng He, Chris Ding, Horst Simon, and Ming Gu. Bipartite graph partitioning and data clustering. In *Proceedings of the tenth international conference on Information and knowledge management*, CIKM '01, pages 25–32, New York, NY, USA, 2001. ACM.
9. Erik D. Demaine, Dotan Emanuel, Amos Fiat, and Nicole Immorlica. Correlation clustering in general weighted graphs. *Theoretical Computer Science*, 2006.
10. Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. Clustering with qualitative information. *J. Comput. Syst. Sci.*, 71(3):360–383, 2005.
11. Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: Ranking and clustering. *J. ACM*, 55(5):1–27, 2008.
12. Nir Ailon and Edo Liberty. Correlation clustering revisited: The "true" cost of error minimization problems. In *ICALP '09: Proceedings of the 36th International Colloquium on Automata, Languages and Programming*, pages 24–36, Berlin, Heidelberg, 2009. Springer-Verlag.
13. Anke van Zuylen and David P. Williamson. Deterministic pivoting algorithms for constrained ranking and clustering problems. *Math. Oper. Res.*, 34(3):594–620, 2009. Preliminary version appeared in SODA'07 (with Rajneesh Hegde and Kamal Jain).
14. Ioannis Giotis and Venkatesan Guruswami. Correlation clustering with a fixed number of clusters. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete Algorithms(SODA)*, pages 1167–1176, New York, NY, USA, 2006. ACM.
15. Marek Karpinski and Warren Schudy. Linear time approximation schemes for the gale-berlekamp game and related minimization problems. *CoRR*, abs/0811.3244, 2008.

A A Counter Example for a Previously Claimed Result

In [1] the authors claim to design and analyze a 4-approximation algorithm for BCC. Its analysis is based on bad squares (and not unbounded structures, as

done in our analysis). Their algorithm is as follows: First, choose a pivot node uniformly at randomly from the left side, and cluster it with all its neighbors. Then, for each node on the left, if it has a neighbor in the newly created cluster, append it with probability 1/2. An exception is reserved for nodes whose neighbor list is identical that of the pivot, in which case these nodes join with probability 1. Remove the clustered nodes and repeat until no nodes are left in the graph.

Unfortunately, there is an example demonstrating that the algorithm has an unbounded approximation ratio. Consider a bipartite graph on $2n$ nodes, $\ell_{1,\dots,n}$ on the left and $r_{1,\dots,n}$ on the right. Let each node ℓ_i on the left be connected to all other nodes on the right except for r_i. The optimal clustering of this graph connects all ℓ_i and r_i nodes and thus has cost $OPT = n$. In the above algorithm, however, the first cluster created will include all but one of the nodes on the right and roughly half the left ones. This already incurs a cost of $\Omega(n^2)$ which is a factor n worse than the best possible.

B Proof of Theorem 1

Proof. Theorem 3.1 in [13] shows that, if the following two conditions hold at the start of the algorithm:

(i) (i) $w^-_{ij} \le 4c_{ij}$ for all $(i,j) \in E^+$, and $w^+_{ij} \le 4c_{ij}$ for all $(i,j) \in E^-$, and
(ii) (ii) $w^+_{ij} + w^+_{jk} + w^-_{ki} \le 4(c_{ij} + c_{jk} + c_{ki})$ for every $(i,j,k) \in \mathcal{B}$, where (k,i) is the unique edge in $E^- \cap \{(i,j),(j,k),(k,i)\}$,

then QuickCluster finds a clustering such that the sum of w^+_{ij} for pairs i,j in different clusters plus the sum of w^-_{ij} for pairs in the same cluster is at most $4\sum_{(i,j)} c_{ij} = \sum_{(i,j)} \left(w^+_{ij} y^-_{(ij)} + w^-_{ij} y^+_{ij} \right)$.

Since $w^+_{ij} = 1$ only if $(i,j) \in E$, and $w^-_{ij} = 1$ only if $(i,j) \in (L \times R) \setminus E$, QuickCluster, in fact, finds a clustering with a number of violated pairs which is at most four times the objective of $\mathrm{LP}_{\mathrm{det}}$. It thus remains to verify that the two conditions (i) and (ii) hold.

It is easy to see that the first condition holds, since if $(i,j) \in E^+$ (resp. E^-) then $y^+_{ij} \ge \frac{1}{2}$ (resp. $y^-_{ij} \ge \frac{1}{2}$), and hence $c_{ij} \ge \frac{1}{2} w^-_{ij}$ (resp. $c_{ij} \ge \frac{1}{2} w^+_{ij}$).

For any bad triplet $(i,j,k) \in \mathcal{B}$ for which all vertices are in L or R, the lefthand side of the second condition is zero and hence the condition holds. Otherwise, we have that either i,j are on the same side of the original bipartite graph, or i,k are on the same side.

In the first case, $w^+_{ij} + w^+_{jk} + w^-_{ki} = w^+_{jk} + w^-_{ki}$. Note that this is at most two, hence it suffices to show that $c_{ij} + c_{jk} + c_{ki} \ge \frac{1}{2}$. Note that if $w^+_{jk} = 0$, then $c_{jk} = w^-_{jk} y^+_{jk} \ge \frac{1}{2}$, since $w^-_{jk} = 1$ and, because $(j,k) \in E^+$, $y^+_{jk} \ge \frac{1}{2}$. Similarly, if $w^-_{ki} = 0$, we have $c_{ki} \ge \frac{1}{2}$. Finally, if $w^+_{jk} = w^-_{ki} = 1$, then $c_{ij} + c_{jk} + c_{ki} = y^-_{jk} + y^+_{ki}$. By the constraints of the linear program, this is at least $1 - y^-_{jk} = y^+_{jk}$, which in turn is at least $\frac{1}{2}$ because $(j,k) \in E^+$.

If i, k are on the same side of the original bipartite graph, the argument is similar: We have that $w^+_{ij} + w^+_{jk} + w^-_{ki} = w^+_{ij} + w^+_{jk}$. Again, if one of w^+_{ij}, w^+_{jk} is zero, then either $c_{ij} \geq \frac{1}{2}$ or $c_{jk} \geq \frac{1}{2}$. If they are both one, then $c_{ij} + c_{jk} + c_{ki} = y^-_{ij} + y^-_{jk} \geq 1 - y^+_{ik} = y^-_{ik}$, and this is at least $\frac{1}{2}$ because $(i,k) \in E^-$. This concludes the proof. □

C Case 2 in Proof of Lemma 3

Here $\alpha_T = \alpha_{\bar{T}} = \min\left\{1, \frac{|R^T_{1,2}|}{|R^T_1| + |R^T_{1,2}|}\right\}$, therefore,

$$\begin{aligned}\beta(T) + \beta(\bar{T}) &= \alpha_T \cdot q_T \cdot \left(\min\{|R^T_{1,2}|, |R^T_2|\} + \min\{|R^T_{1,2}|, |R^T_1|\}\right) \\ &= q_T \cdot \min\{|R^T_{1,2}| + |R^T_1|, |R^T_{1,2}|\} = q_T \cdot |R^T_{1,2}|.\end{aligned}$$

As $|R^T_1| \leq |R^T_{1,2}|$, if event X_T happens PivotBiCluster adds ℓ^T_2 to ℓ^T_1's cluster with probability $\min\left\{\frac{|R^T_{1,2}|}{|R^T_2|}, 1\right\} = \frac{|R^T_{1,2}|}{|R^T_2|}$. Therefore with probability $\frac{|R^T_{1,2}|}{|R^T_2|}$ the pairs colored by color T that PivotBiCluster violate are all the edges from ℓ^T_2 to R^T_2 and all the non-edges from ℓ^T_2 to R^T_1, and with probability $\left(1 - \frac{|R_{1,2}|}{|R_2|}\right)$ PivotBiCluster violates all the edges from ℓ^T_2 to $R^T_{1,2}$. Thus,

$$\begin{aligned}\mathbb{E}[\text{cost}(T)|X_T] &= \frac{|R^T_{1,2}|}{|R^T_2|}\left(|R^T_2| + |R^T_1|\right) + \left(1 - \frac{|R^T_{1,2}|}{|R^T_2|}\right)|R^T_{1,2}| \\ &= 2 \cdot |R^T_{1,2}| + \frac{|R^T_{1,2}| \cdot |R^T_1| - |R^T_{1,2}|^2}{|R^T_2|} \;\leq\; 2 \cdot |R^T_{1,2}|.\end{aligned}$$

If the event $X_{\bar{T}}$ happens, as $|R^{\bar{T}}_1| > |R^{\bar{T}}_{1,2}|$ and $\min\left\{\frac{R^{\bar{T}}_{1,2}}{R^{\bar{T}}_2}, 1\right\} = 1$, PivotBiCluster chooses to isolate $\ell^{\bar{T}}_2$ $(= \ell^T_1)$ almost surely and the number of pairs colored with color $\bar{T}$ that are consequently violated is $|R^{\bar{T}}_2| + |R^{\bar{T}}_{1,2}| = |R^T_1| + |R^T_{1,2}|$. Thus,

$$\begin{aligned}q_T \cdot \left(\mathbb{E}[\text{cost}(T)|X_T]) + \mathbb{E}[\text{cost}(\bar{T})|X_{\bar{T}})]\right) &\leq q_T \cdot (2|R^T_{1,2}| + |R^T_1| + |R^T_{1,2}|) \\ &< 4 \cdot q_T \cdot |R^T_{1,2}| = 4 \cdot \left(\beta(T) + \beta(\bar{T})\right) \;.\end{aligned}$$

D Case 3 in Proof of Lemma 3

Here, $\alpha_T = \alpha_{\bar{T}} = \frac{1}{2}$, thus,

$$\beta(T) + \beta(\bar{T}) = \frac{1}{2} \cdot q_T \cdot \left(\min\{|R^T_{1,2}|, |R^T_2|\} + \min\{|R^T_{1,2}|, |R^T_1|\}\right) = q_T \cdot |R^T_{1,2}| \;.$$

Conditioned on event X_T, as $|R^T_1| > |R^T_{1,2}|$, PivotBiCluster chooses to isolate ℓ_2 with probability $\min\left\{\frac{|R^T_{1,2}|}{|R^T_2|}, 1\right\} = \frac{|R^T_{1,2}|}{|R^T_2|}$. Therefore with probability $\frac{|R^T_{1,2}|}{|R^T_2|}$

PivotBiCluster colors $|R_2^T| + |R_{1,2}^T|$ pairs with color T (and violated them all). With probability $\left(1 - \frac{|R_{1,2}^T|}{|R_2^T|}\right)$, PivotBiCluster colors $|R_{1,2}^T|$ pairs with color T (and violated them all). We conclude that

$$\mathbb{E}[\text{cost}(T)|X_t] = \frac{|R_{1,2}^T|}{|R_2^T|}(|R_2^T| + |R_{1,2}^T|) + \left(1 - \frac{|R_{1,2}^T|}{|R_2^T|}\right)|R_{1,2}^T| = 2|R_{1,2}^T| \ .$$

Similarly, for event $X_{\bar{T}}$, as $|R_1^{\bar{T}}| > |R_{1,2}^{\bar{T}}|$ and $\min\left\{\frac{|R_{1,2}^{\bar{T}}|}{|R_2^{\bar{T}}|}, 1\right\} = \frac{|R_{1,2}^T|}{|R_1^T|}$, PivotBiCluster isolates ℓ_1 with probability $\frac{|R_{1,2}^T|}{|R_1^T|}$ therefore colors $|R_2^{\bar{T}}| + |R_{1,2}^{\bar{T}}|$ pairs with color $\bar{T}$ (and violated them all). With probability $(1 - \frac{|R_{1,2}^T|}{|R_1^T|})$ PivotBiCluster colors $|R_{1,2}^{\bar{T}}|$ pairs with color $\bar{T}$ (and violates them all). Thus,

$$\mathbb{E}[\text{cost}(\bar{T})|X_{\bar{T}}] = \frac{|R_{1,2}^T|}{|R_1^T|}(|R_1^T| + |R_{1,2}^T|) + \left(1 - \frac{|R_{1,2}^T|}{|R_1^T|}\right)|R_{1,2}^T| = 2|R_{1,2}^T| \ .$$

Hence, $q_T \cdot \left(\mathbb{E}[\text{cost}(T)|X_t] + \mathbb{E}[\text{cost}(\bar{T})|X_{\bar{T}}]\right) = 4 \cdot q_T \cdot |R_{1,2}^T| = 4 \cdot (\beta(T) + \beta(\bar{T}))$.
□