A. Additional Experiments

In this section, we provide various statistics regarding the performance and solutions produced by the algorithms. Table 4 presents the number of MPC rounds required by each algorithm, the number of clusters in each solution and the number of existing intra-cluster edges.

Table 4: This table presents the number of MPC rounds (#rounds), number of clusters (#clusters) and the fraction of intra-cluster edges found in each solution. We observe that \textsc{OurAlgo} requires a fixed number of MPC rounds that is significantly smaller (up to a factor 90) compared to \textsc{ClusterW} and \textsc{Pivot}. Moreover, while \textsc{OurAlgo} produces solutions with more clusters compared to \textsc{ClusterW} and \textsc{Pivot}, the produced clusters are much denser than those produced by \textsc{ClusterW} and \textsc{Pivot}.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>#rounds</th>
<th>#clusters</th>
<th>in-edges</th>
<th>#rounds</th>
<th>#clusters</th>
<th>in-edges</th>
<th>#rounds</th>
<th>#clusters</th>
<th>in-edges</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textsc{OurAlgo}-0.05</td>
<td>33</td>
<td>723,511</td>
<td>1.000</td>
<td>33</td>
<td>22,999,216</td>
<td>0.955</td>
<td>33</td>
<td>36,467,636</td>
<td>0.972</td>
</tr>
<tr>
<td>\textsc{OurAlgo}-0.1</td>
<td>33</td>
<td>720,229</td>
<td>0.999</td>
<td>33</td>
<td>22,764,081</td>
<td>0.933</td>
<td>33</td>
<td>34,244,835</td>
<td>0.957</td>
</tr>
<tr>
<td>\textsc{OurAlgo}-0.2</td>
<td>33</td>
<td>704,489</td>
<td>0.996</td>
<td>33</td>
<td>22,228,865</td>
<td>0.895</td>
<td>33</td>
<td>31,042,932</td>
<td>0.735</td>
</tr>
<tr>
<td>\textsc{ClusterW}-0.9</td>
<td>725</td>
<td>382,491</td>
<td>0.516</td>
<td>1441</td>
<td>12,778,648</td>
<td>0.461</td>
<td>1837</td>
<td>22,457,586</td>
<td>0.287</td>
</tr>
<tr>
<td>\textsc{Pivot}-0.9</td>
<td>1160</td>
<td>386,275</td>
<td>0.537</td>
<td>2280</td>
<td>12,944,056</td>
<td>0.452</td>
<td>2610</td>
<td>22,675,174</td>
<td>0.316</td>
</tr>
</tbody>
</table>

B. Missing Proofs from Section 3

B.1. Proof of Fact 3.2

Proof of (1). Without loss of generality, assume that \(d(u) \leq d(v)\). We have \(|N(u) \triangle N(v)| \geq d(v) - d(u)\). Then, by Definition 3.1, \(d(v) - d(u) \leq |N(u) \triangle N(v)| \leq i \beta \cdot d(v)\). This now implies \(d(u) \geq (1 - i \beta)d(v)\), as desired.

Proof of (2). For \(i = 1, ..., k - 1\), we have by (1):

\[
d(v_i) \leq \frac{d(v_{i+1})}{1 - \beta} \leq \ldots \leq \frac{d(v_k)}{(1 - \beta)^{k-1}} \leq \frac{d(v_k)}{(1 - \beta)^2} \leq \frac{k}{k - 1} \cdot d(v_k),
\]

since \((1 - \beta)^4 \geq (1 - \frac{1}{n})^4 > \frac{4}{9} \geq \frac{k-1}{k}^3\). Now we iterate the triangle inequality:

\[
|N(v_1) \triangle N(v_k)| \leq \sum_{i=1}^{k-1} |N(v_i) \triangle N(v_{i+1})| \\
< \sum_{i=1}^{k-1} \beta \cdot \max(d(v_i), d(v_{i+1})) \\
\leq (k - 1) \cdot \beta \cdot \frac{k}{k - 1} \cdot d(v_k) \\
\leq k \cdot \beta \cdot \max(d(v_1), d(v_k)).
\]
Proof of (3). Without loss of generality, assume that \( d(u) \leq d(v) \). Then
\[
|N(u) \cap N(v)| = |N(v)| - |N(v) \setminus N(u)|
\geq |N(v)| - |N(u) \triangle N(v)|
\geq (1 - i\beta)d(v).
\]

B.2. Proof of Lemma 3.3

Lemma 3.3. Suppose that \( 5\beta + 2\lambda < 1 \). Let \( CC \) be a connected component of \( \tilde{G} \). Then, for every \( u, v \in CC \):

(a) if \( u \) and \( v \) are heavy, then \( \text{dist}\tilde{G}(u, v) \leq 2 \),
(b) $\text{dist}^G(u, v) \leq 4$,
(c) $\text{dist}^G(u, v) \leq 2$,
(d) if $u$ or $v$ is heavy, then $u$ and $v$ are in 4-weak agreement.

**Proof.** For (a), suppose by contradiction that there are heavy $u, v \in CC$ with $\text{dist}^G(u, v) > 2$; pick such $u, v$ with minimum $\text{dist}^G(u, v)$. If $\text{dist}^G(u, v) \geq 5$, let $P = \langle u, u', u'', ..., v \rangle$ be a shortest $u$-$v$ path in $\tilde{G}$; since there are no edges in $\tilde{G}$ with both endpoints being light, either $u'$ or $u''$ must be heavy, and the pair $(u', v)$ or $(u'', v)$ contradicts the minimality of the path $(u, v)$ (as we have $\text{dist}^G(u', v) > 2$).

On the other hand, if $\text{dist}^G(u, v) \leq 4$, then by Fact 3.2 $(2)$, $u$ and $v$ are in weak agreement, and by Fact 3.2 $(3)$, we have $|N(u) \setminus N(v)| \geq (1 - 5\beta)d(v)$. Note that a heavy vertex can lose at most a $\lambda$ fraction of its neighbors in $G$ in Line 1 of the algorithm, and it loses no neighbors in Line 3, thus $|N(v) \setminus N^G(v)| \leq \lambda d(v)$ and similarly for $u$. Assume without loss of generality that $d(v) \geq d(u)$. Then we have

$$|\tilde{N}(u) \cap \tilde{N}(v)| \geq |N(u) \cap N(v)| - |N(u) \setminus \tilde{N}(u)| - |N(v) \setminus \tilde{N}(v)| \geq (1 - 5\beta - 2\lambda)d(v) > 0,$$

i.e., $u$ and $v$ have a common neighbor in $\tilde{G}$, and thus, $\text{dist}^G(u, v) \leq 2$.

For (b), let $P$ be a shortest $u$-$v$ path in $\tilde{G}$. Define the vertex $u'$ to be $u$ if $u$ is heavy and to be $u$’s neighbor on $P$ if $u$ is light; in the latter case, $u'$ is heavy since there are no edges in $\tilde{G}$ with both endpoints being light. Define $v'$ similarly. Since $u'$ and $v'$ are heavy, we have $\text{dist}^G(u, v) \leq 1 + \text{dist}^G(u', v') + 1 \leq 4$.

For (c), note that by (b) and Fact 3.2 $(2)$, $u$ and $v$ are in weak agreement; by Fact 3.2 $(3)$, they have at least $(1 - 5\beta)d(v) > 0$ common neighbors in $G$.

To prove (d), we proceed similarly as for (b). We consider two cases: both $u$ and $v$ are heavy; only one $u$ or $v$ is heavy. In the first case, by (a) and Fact 3.2 $(2)$, we even have that $u$ and $v$ are in 3-weak agreement. In the second case, one of the vertices is light; without loss of generality, assume $u$ is light. In that case, $u$ is adjacent to a heavy vertex $u'$, as there are no edges between light vertices. Since by (a) $v$ and $u'$ are at distance 2, it implies that $v$ and $u$ are at distance 3. Since each edge $(x, y)$ in $CC$ means that $x$ and $y$ are in agreement, by Fact 3.2 $(2)$, we have that $v$ and $u$ are in 4-weak agreement. □

### B.3. Proof of Lemma 3.4

**Lemma 3.4.** Let $CC$ be a connected component of $\tilde{G}$ such that $|CC| \geq 2$. Then, for each vertex $u \in CC$ we have that

$$d(u, CC) \geq (1 - 8\beta - \lambda)|CC|.$$

**Proof.** Assume that $CC$ is a non-trivial connected component, i.e., $CC$ has at least two vertices. Let $x$ be a heavy vertex in $CC$. Observe that such a vertex $x$ always exists by the construction of our algorithm; edges having both light endpoints are removed in Line 3 of **Algorithm 1**.

**Remark:** While $CC$ refers to a connected component in the sparsified graph $\tilde{G}$, note that $N(\cdot)$ and $d(\cdot)$ refer to neighborhood and degree functions with respect to the input graph $G$ rather than with respect to $\tilde{G}$.

First, from Lemma 3.3 $(4)$, we have that any two vertices in $CC$, one of which is heavy, are in 4-weak agreement. In particular, this also holds for $x$ and any other vertex $u \in CC$. As defined in Section 2, recall that $N(x, CC) \triangleq N(x) \cap CC$. Since $x$ is a heavy vertex, it has at most a $\lambda$-fraction of its neighbors $N(x)$ outside $CC$, and so from Fact 3.2 $(3)$, we have

$$|N(x, CC) \cap N(u)| \geq (1 - 4\beta)d(x) - \lambda d(x) = (1 - 4\beta - \lambda)d(x). \tag{2}$$

Observe that this also implies

$$|N(u, CC)| \geq (1 - 4\beta - \lambda)d(x). \tag{3}$$

Next, we want to upper-bound the number of vertices in $CC \setminus N(x)$, which will enable us to express $|CC|$ as a function of $d(x)$. To that end, note that Equation $(2)$ implies a lower bound on the number of edges between the neighbors of $x$ in $CC$, denoted by $N(x, CC)$, and the vertices in $CC$ other than $N(x)$, denoted by $CC \setminus N(x)$, as follows:

$$|E(N(x, CC), CC \setminus N(x))| \geq |CC \setminus N(x)| \cdot (1 - 4\beta - \lambda)d(x), \tag{4}$$
where \( E(Y, Z) \) is the set of edges between sets \( Y \) and \( Z \). On the other hand, since \( d(u) \leq \frac{d(x)}{1 - 4\beta} \) for each \( u \in CC \) by Fact 3.2 and since \( u \) and \( x \) are in 4-weak agreement, we have that \( u \) has at most \( 4\beta \frac{d(x)}{1 - 4\beta} \) neighbors outside \( N(x) \). Hence, we derive
\[
|E(N(x, CC), CC \setminus N(x))| \leq |N(x, CC)| \cdot \frac{4\beta d(x)}{1 - 4\beta} \leq d(x) \cdot \frac{4\beta d(x)}{1 - 4\beta}.
\]
Combining the last inequality with Equation (4) yields
\[
|CC \setminus N(x)| \leq \frac{4\beta d(x)}{(1 - 4\beta) \cdot (1 - 4\beta - \lambda)} \leq \frac{4\beta d(x)}{1 - 8\beta - \lambda},
\]
which further implies
\[
|CC| = |CC \setminus N(x)| + |N(x, CC)| \leq \left(1 + \frac{4\beta}{1 - 8\beta - \lambda}\right) d(x) = \frac{1 - 4\beta - \lambda}{1 - 8\beta - \lambda} d(x).
\]
Now together with Equation (3) we establish
\[
|N(u, CC)| \geq (1 - 8\beta - \lambda)|CC|,
\]
as desired.

\[\square\]

**B.4. Proof of Lemma 3.5**

**Lemma 3.5.** Let \( CC \) be a connected component in \( \tilde{G} \). Assume that \( 8\beta + \lambda \leq 1/4 \). Then, the cost of keeping \( CC \) as a cluster in \( G \) is no larger than the cost of splitting \( CC \) into two or more clusters.

**Proof.** Towards a contradiction, consider a split of \( CC \) into \( k \geq 2 \) clusters \( C_1, \ldots, C_k \) whose cost is less than the cost of keeping \( CC \) as a single cluster. Moreover, consider the cheapest such split of \( CC \). Let \( \delta \equiv 8\beta + \lambda \). We consider two cases: when each cluster in \( \{C_1, \ldots, C_k\} \) has size at most \((1 - 2\delta)|CC|\) vertices, and the complement case.

It holds that \(|C_i| \leq (1 - 2\delta)|CC|\) for each \( i \). By Lemma 3.4, each vertex \( v \in C_i \) for each cluster \( C_i \) has at least \((1 - \delta)|CC| - |C_i| \geq \delta|CC|\) neighbors in \( CC \setminus C_i \). Hence, splitting \( CC \) in the described way cuts at least \( \frac{\delta|CC|^2}{2} \) “+” edges. On the other hand, also by Lemma 3.4 \( CC \) has at most \( \frac{\delta|CC|^2}{2} \) “-” edges. Hence, it does not cost less to split \( CC \) in the described way.

There exists a cluster \( C^* \) such that \(|C^*| > (1 - 2\delta)|CC|\). Let \( C_i \neq C^* \) be one of the clusters \( CC \) is split into. Clearly, we have \(|C_i| < 2\delta|CC|\). Since, by Lemma 3.4 each vertex \( v \in C_i \) has at least \((1 - \delta)|CC| \) “+” edges inside \( CC \), it implies that \( v \) has more than \((1 - 3\delta)|CC| \) “+” edges to \( C^* \). On the other hand, there are at most \( \delta|CC| \) “-” edges from \( v \) to \( C^* \). Hence, as long as \( 1 - 3\delta \geq \delta \), it implies that it is cheaper to merge \( C^* \) with \( C_i \) than to keep them split. This contradicts our assumption that the split into those \( k \) clusters results in the minimum cost.

Observe that the condition \( 1 - 3\delta \geq \delta \) is equivalent to \( 8\beta + \lambda \leq 1/4 \), which holds by our assumption.

\[\square\]

**B.5. Proof of Lemma 3.6**

**Lemma 3.6.** Let \( G' \) be a non-complete graph obtained from \( G \) by removing any “+” edge \( \{u, v\} \) (i.e., changing it into a “neutral” edge) where \( u \) and \( v \) belong to different connected components of \( \tilde{G} \). Then, our algorithm outputs a solution that is optimal for the instance \( G' \).

**Proof.** It is suboptimal for a single cluster to contain vertices from different connected components; indeed, breaking such a cluster up into connected components would improve the objective function (all edges between connected components are negative). Therefore any optimal solution must either be equal to our solution or it should split some cluster in our solution. The claim follows, by Lemma 3.5 because subdividing a connected component of \( G' \) (equivalently of \( \tilde{G} \)) does not improve the objective function.

\footnote{We remark that everywhere else in the paper, correlation clustering instances are always complete graphs.}
We are left with verifying property (2). Fix an edge/non-edge where $a$ belongs to the same cluster in $O$. Then, for each $w \in |N(u) \cap N(v)|$, $O$ pays for one of the edges/non-edges $(u, w), (v, w)$. If $w$ is in the same cluster as $u, v$, then $O$ pays for the one of $(u, w), (v, w)$ that is a non-edge; and vice versa. So $(u, v)$ can assign $\frac{1}{\beta \max(d(u), d(v))}$ units of debt to that edge/non-edge. This way, properties (1) and (3) are clear.

We verify property (2). Fix an edge/non-edge $(a, b)$ that $O$ pays for. It is only charged by adjacent edges. Each edge adjacent to $a$, of which there are $d(a)$ many, assigns at most $\frac{1}{\beta \cdot d(a)}$ units of debt; this gives $\frac{2}{\beta}$ units in total. The same holds for edges adjacent to $b$; together this yields $\frac{2}{\beta}$ units.

B.7. Proof of Lemma 3.8

**Lemma 3.8.** The number of edges deleted in Line 7 of our algorithm that are not cut in $O$ is at most $\left(\frac{1}{\beta} + \frac{1}{\lambda} + \frac{1}{\lambda\cdot d(y)}\right) \cdot OPT$.

**Proof.** We use a similar charging argument as in the proof of Lemma 3.7, with the difference that each edge/non-edge that $O$ pays for will be assigned at most $\frac{1}{\beta \cdot d(a)}$ units of debt (rather than at most $\frac{2}{\beta}$).

Let $(u, v)$ be an edge as in the statement. For each endpoint $y \in \{u, v\}$, we proceed as follows. As $y$ is light, there are edges $(y, v_1), ..., (y, v_{\lambda \cdot d(y)})$ whose endpoints are not in agreement. For each $i = 1, ..., \lambda \cdot d(y)$, proceed as follows:

- If $(y, v_i)$ is not cut by $O$, then, as in the proof of Lemma 3.7, $(y, v_i)$ has at least $\beta \cdot \max(d(y), d(v_i))$ adjacent edges/non-edges for whom $O$ pays. Each of these edges/non-edges is of the form $(v_i, w)$ or $(y, w)$. We will have the edge $(u, v)$ charge $\frac{1}{2 \beta \lambda d(y_1) \max(d(y), d(v_i))}$ units of debt, which we will call blue debt, to the former ones (those of the form $(v_i, w)$), and $\frac{1}{2 \beta \lambda d(y)}$ units of debt, which we will call red debt, to the latter ones (those of the form $(y, w)$).\(^6\)

- If $(y, v_i)$ is cut by $O$, then $O$ pays for $(y, v_i)$. We will have the edge $(u, v)$ charge $\frac{1}{2 \beta \lambda d(y)}$ units of debt, which we will call green debt, to $(y, v_i)$.

Let us verify property (1). In the first case, each of these edges/non-edges is charged at least $\frac{1}{2 \beta \lambda d(y_1) \max(d(y), d(v_i))}$ units of debt, and since there are at least $\beta \cdot \max(d(y), d(v_i))$ of them, the total (blue or red) debt charged is at least $\frac{1}{2 \beta \lambda d(y)}$ per each $y \in \{u, v\}$ and each $i = 1, ..., \lambda \cdot d(y)$. This much total (green) debt is also charged in the second case. Since there are $2$ choices for $y$ and then $\lambda \cdot d(y)$ choices for $i$, in total the edge $(u, v)$ assigns at least $1$ unit of debt. Property (3) is satisfied by design.

We are left with verifying property (2). Fix an edge/non-edge $(a, b)$ that $O$ pays for. It can be charged by its adjacent edges (red or green debt), as well as those at distance two (blue debt). Let us consider these cases separately.

**Adjacent edges (red/green debt):** let us first look at edges adjacent to $a$ (we will get half of the final charge this way). That is, $a$ is serving the role of $y$ above; it can serve that role for at most $d(a)$ debt-charging edges (serving the role of $(u, v)$, where $a = y \in \{u, v\}$).

- **Red debt:** each of these debt-charging edges charges $(a, b)$ at most $\lambda \cdot d(a)$ times (once per $i = 1, ..., \lambda \cdot d(y)$), and each charge is for $\frac{1}{2 \beta \lambda d(a) \cdot \max(d(y), d(v_i))}$ units of debt. This gives $\frac{1}{2 \beta \lambda d(a)} \cdot \lambda d(a) = \frac{1}{\beta \cdot d(a)}$ units of debt.

- **Green debt:** each of these debt-charging edges charges $(a, b)$ at most once (if it happens that $(a, b) = (y, v_i)$ for some $i$), and each charge is for $\frac{1}{\max(d(y), d(v_i))}$ units of debt. This gives $\frac{1}{\beta \cdot d(a)}$ units of debt.\(^6\)

Notice that the latter edges/non-edges might be charged many times by the same $y$ (for different $i$).
We get the same amount from edges adjacent to $b$ ($b$ serving the role of $y$). In total, we get a debt of $\frac{1}{\beta} + \frac{1}{X}$.

**Blue debt:** $(a, b)$ is serving the role of $(v_i, w)$ above. Let us first look at $a$ serving the role of $v_i$ (we will get half of the final charge this way). Then a neighbor of $a$ must be serving the role of $y$. There are at most $d(a)$ possible $y$’s, and at most $d(y)$ possible edges $(u, v)$ for each $y$ (those with $y \in \{u, v\}$). Recall that each charge was for $\frac{1}{2\beta d(a) d(y)}$ units of debt; per $y$, this sums up (over edges $(u, v)$) to at most $\frac{1}{2\beta d(a) d(y)} \cdot d(y) = \frac{1}{2\beta d(a)}$ total units, and since there are at most $d(a)$ many $y$’s, the total debt is at most $\frac{1}{2\beta X}$. We get the same amount from $b$ serving the role of $v_i$. In total, we get a debt of $\frac{1}{\beta X}$.

**B.8. Proof of Remark 3.10**

**Remark 3.10.** For fixed values of $\beta$ and $\lambda$, the above analysis is tight, in the sense that the term $\frac{1}{\beta X}$ is necessary.

**Proof.** Let us assume for simplicity that $\beta = \lambda$; otherwise the example can be adapted. Consider the following instance: two disjoint cliques $A_1, A_2$ of size $(1 - \beta)d$ each, with a subset $X_1 \subseteq A_1$ and a subset $X_2 \subseteq A_2$, both of size $\beta d$, fully connected to each other.

The optimal solution is to have two clusters ($A_1$ and $A_2$). The cost is $(\beta d)^2$ (cutting the edges between $X_1$ and $X_2$).

However, our algorithm will first delete the edges between $A_1 \setminus X_1$ and $X_1$ (any two vertices from these respective sets are not in agreement, as the $X_1$-vertex has $\beta d$ extra neighbors in $X_2$), between $X_1$ and $X_2$, and between $A_2 \setminus X_2$ and $X_2$.

Then every vertex in the graph becomes light. Thus in Line 3 we delete all edges, making $\tilde{G}$ an empty graph. Finally, we return the singleton partitioning as the solution. Its cost is $(\beta d)^2 + 2 \cdot (d(1 - \beta))^2 \approx \left(\frac{1}{\beta X} - \frac{2}{\beta} + 2\right) \cdot \text{OPT}$. 

**B.9. Proof of Lemma 3.11**

**Lemma 3.11.** For any constant $\delta > 0$, there exists an MPC algorithm that, given a signed graph $G = (V, E^\pm)$, in $O(1)$ rounds for all pairs of vertices $(u, v) \in E^+$ outputs “Yes” if $u$ and $v$ are in $0.68$-weak agreement, and outputs “No” if $u$ and $v$ are not in agreement. Letting $n = |V|$, this algorithm succeeds with probability $1 - 1/n$, uses $n^\delta$ memory per machine, and uses a total memory of $O(|E^+|)$.

To prove Lemma 3.11 we will use the following well-known concentration inequalities.

**Theorem B.1 (Chernoff bound).** Let $X_1, \ldots, X_k$ be independent random variables taking values in $[0, 1]$. Let $X \overset{\text{def}}{=} \sum_{i=1}^k X_i$. Then, the following inequalities hold:

(a) For any $\delta \in [0, 1]$ if $\mathbb{E}[X] \leq U$ we have

$$
\mathbb{P}[X \geq (1 + \delta)U] \leq \exp \left(-\delta^2 U/3\right).
$$

(b) For any $\delta > 0$ if $\mathbb{E}[X] \geq U$ we have

$$
\mathbb{P}[X \leq (1 - \delta)U] \leq \exp \left(-\delta^2 U/2\right).
$$

**Lemma B.2.** Let $u$ and $v$ be two vertices. If Algorithm 2 returns “Yes”, then for $\alpha \geq 600$ with probability at least $(1 - n^{-3})$ it holds that $u$ and $v$ are in agreement. (Conversely, the algorithm outputs “No” with probability at least $(1 - n^{-3})$ if $u$ and $v$ are not in agreement.)

**Proof.** We now upper-bound the probability that $u$ and $v$ are not in agreement, but Algorithm 2 returns “Yes”.

Assume that $u$ and $v$ are not in agreement. Then

$$
\mathbb{E}[X_{u,v}] > \tau,
$$

where $\tau$ is defined in Algorithm 2 (As a reminder, $X_{u,v}$ is defined in Equation (1)). Algorithm 2 passes the test on Line 5 with probability

$$
\mathbb{P}[X_{u,v} \leq 0.9\tau] \overset{\text{Theorem B.1(a)}}{\leq} \exp \left(-1/100 \cdot \frac{\alpha \cdot \log n}{2}\right).
$$

\(^7\)As an aside, note that by now, the algorithm has paid around $\left(1 + \frac{2}{\beta}\right) \cdot \text{OPT}$, showing that Lemma 3.7 by itself is also tight for Line 1.
where we used that $d(u)/j \geq 1$. For $a \geq 600$, the last expression is upper-bounded by $n^{-3}$.

**Lemma B.3.** Let $u$ and $v$ be two vertices that are in 0.8-weak agreement. Then, for $a \geq 600$ with probability at least $(1 - n^{-3})$, Algorithm 2 outputs “Yes”.

**Proof.** We have

$$E[X_{u,v}] \leq 0.8 \cdot \tau,$$

where $\tau$ is defined in Algorithm 2. Hence, Algorithm 2 outputs “No” with probability

$$P[X_{u,v} > 0.9 \cdot \tau] \leq \exp \left( -\frac{1}{64} \cdot \frac{a \cdot \log n}{3} \right),$$

where we used that $d(u)/j \geq 1$. For $a \geq 600$, the last expression is upper-bounded by $n^{-3}$.

The implementation part of Lemma 3.11 follows by our discussion in Section 3.2 and by having $a = O(1)$. The claim on probability success follows by using Lemmas B.2 and B.3 and applying a union bound over all $|E^+| \leq n^2$ pairs of vertices.