Online and Consistent Correlation Clustering

Vincent Cohen-Addad [*1] Silvio Lattanzi [*1] Andreas Maggiori [*2] Nikos Parotsidis [*1]

Abstract

In the correlation clustering problem the input is a signed graph where the sign indicates whether each pair of points should be placed in the same cluster or not. The goal of the problem is to compute a clustering which minimizes the number of disagreements with such recommendation. Thanks to its many practical applications, correlation clustering is a fundamental unsupervised learning problem and has been extensively studied in many different settings. In this paper we study the problem in the classic online setting with recourse; The vertices of the graphs arrive in an online manner and the goal is to maintain an approximate clustering while minimizing the number of times each vertex changes cluster. Our main contribution is an algorithm that achieves logarithmic recourse per vertex in the worst case. We also complement this result with a tight lower bound. Finally we show experimentally that our algorithm achieves better performances than state-of-the-art algorithms on real world data.

1. Introduction

Clustering is a fundamental problem in unsupervised learning. In clustering one is interested in partitioning the input elements so that similar elements are grouped together and different elements are assigned to different clusters. A natural way to capture this notion is the classic correlation clustering problem. Thanks to its simple and elegant formulation, the correlation clustering problem received a lot of attention from the theory and applied communities and it has found many practical applications including finding clustering ensembles (Bonchi et al., 2013), duplicate detection (Arasu et al., 2009), community mining (Chen et al., 2012), disambiguation tasks (Kalashnikov et al., 2008), automated labelling (Agrawal et al., 2009; Chakrabarti et al., 2008) and many more.

Formally, in the correlation clustering problem (Bansal et al., 2004) we receive as input a weighted graph, where positive edges represent similarities between nodes and negative edges represent dissimilarities between them. The goal is to find a partitioning of the input graph so that the sum of the weights of the negative edges inside clusters and the positive edges between clusters is minimized. The problem is NP-hard and several approximation algorithms have been proposed for it. In particular, when we focus on the cases where all edges have weights in $\{-1, +1\}$ the best known algorithm (Chawla et al., 2015) has an approximation guarantee of 2.06 while for arbitrary weights a $O(\log n)$ approximation algorithm is also known (Demaine et al., 2006)[1].

Since real world datasets continuously evolve in time, the design of approximation algorithms that maintain a good solution over time is becoming a central question in machine learning. Unfortunately, classic approximation algorithms often are either unpractical or they return very unstable solutions on dynamic datasets. In particular, the clustering returned by the algorithm may drastically change over time, potentially leading to inconsistent decisions. For this reason a lot of attention has been devoted in the past few years in designing efficient and consistent clustering algorithms for evolving datasets (Lattanzi & Vassilvitskii, 2017; Fichtenberger et al., 2021; Jaghargh et al., 2019; Cohen-Addad et al., 2019; Guo et al., 2021). In this paper, we study the correlation clustering problem in the online setting. In this setting, nodes arrive one at the time and with them also their clustering preferences to the previously disclosed nodes are revealed, or in other word the set of positive and negative edges to the nodes that have already arrived. The algorithm has to assign a cluster to every node at its arrival and this assignment cannot change in the course of the algorithm.

Unfortunately this setting is too restrictive and in fact it has been shown (Mathieu et al., 2010) that any algorithm for the online correlation clustering problem has at least $\Omega(n)$ competitive ratio. Intuitively, this is true because if the first

[*]Equal contribution [1]Google [2]EPFL, Lausanne, Switzerland. Correspondence to: Andreas Maggiori <andreas.maggiori@epfl.ch>.

Proceedings of the 39th International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

[1]Note also that there is an important amount of work on the version of the problem where the objective is to maximize the number of positive edges whose both endpoints are in the same cluster plus the number negative edges across clusters

edge that is revealed is a positive edge than one cannot distinguish the case that this edge is part of a large clique or is the only bridge between two cliques. To get beyond this negative result, the problem has been studied in the random setting where Ailon et al. (2008) show that the Pivot algorithm is 3-competitive when vertices arrive in random order and in the semi-online model (Lattanzi et al., 2021) when a random fraction of the instance is revealed in advance and where Pivot still obtain a constant approximation. One main shortcoming of these results is that they make a strong assumption on the arrival order and for that reason they may not provide any guarantee in real world scenarios.

To overcome this limitation, in this paper we consider the correlation clustering problem in the online model with recourse. In this model, nodes and edges are still revealed in an online fashion but the algorithm can re-assign nodes to different clusters after each arrival. The goal in this setting is to minimize the number of cluster re-assignments executed by the algorithm while still maintaining a constant factor approximation. From a practical perspective, this setting is interesting because it captures well the cases where changing the clustering assignment is possible but expensive. This is for example the case when the output clustering is used as input in a machine learning pipeline and so re-assigning points requires re-training. From a theoretical perspective, this setting allows us to study formally the trade-off between stability of the solution and quality of approximation. For this reasons other classic clustering problems were studied in this setting (Lattanzi & Vassilvitskii, 2017; Fichtenberger et al., 2021; Cohen-Addad et al., 2019; Guo et al., 2021).

Our contributions. Our first contribution is to design an algorithm that has logarithmic recourse per node and that maintains a constant approximation to the correlation clustering problem at any point in time. Our algorithm is different from previous algorithms for correlation clustering in the online setting and draws inspiration from a recent result in the setting of parallel algorithms for correlation clustering (Cohen-Addad et al., 2021). From a very high level perspective the main idea behind the algorithm is to track dense structure in the graph in an approximate sense by carefully designing lazy updates of the clustering.

We then present a lower bound showing that any algorithm that maintains a constant approximation to the optimal solution has to incur at least logarithmic recourse per node.

Finally, we complement our theoretical results with an experimental analysis showing that our algorithm produces at the same time solutions that are stable and of high-quality. In particular, when compared with the Pivot algorithm, suggested by previous work (Ailon et al., 2008; Lattanzi et al., 2021) in the classical online setting, we notice that our algorithm produces solutions that are substantially more stable and of higher quality.

2. Problem Definition

We focus on the *disagreements minimization* version of the correlation clustering problem where the input is a signed undirected graph $G = (V, E, s)$ where each edge $e = \{u, v\}$ is assigned a sign $s(e) \in \{+, -\}$ and the goal is to find a partition of the vertexes such that the number of $'-'$ edges inside the same cluster and $'+'$ edges in between clusters is minimized. For simplicity we denote the set of $'+'$ and $'-'$ edges by E^+ and E^- respectively. It is convenient to represent a solution to the problem, namely a clustering of the vertices of the graph, using an *assignment function* $f : V \to \mathbb{Z}$; Then, the *clustering* induced by f is a partition of the nodes $\mathcal{C} = \{C_1, C_2, \dots, C_k\}$ such that two nodes u, v belong to the same partition, i.e., cluster C_i, if and only if $f(u) = f(v)$, or in other words, they are *assigned* the same cluster id. Given a partition function f and a clustering $\mathcal{C}$ induced by f we slightly abuse notation and denote by $f(C)$ the common cluster id of all nodes in cluster $C \in \mathcal{C}$. Hence, the cost of an assignment function or the clustering induced by the latter is equal to:

$$\text{cost}(f) = \sum_{\substack{\{u,v\} \in E^+ \\ f(u) \neq f(v)}} 1 + \sum_{\substack{\{u,v\} \in E^- \\ f(u) = f(v)}} 1$$

In the online setting nodes arrive one at a time, revealing upon arrival all the edges to previously arrived nodes. Let $G = (V, E, s)$ be a signed undirected graph, then an instance of the online correlation clustering problem can be described by a pair $\mathcal{I} = (G, \sigma)$ where G is the *final graph* and σ is an order on the vertices of G: $\sigma = \langle v_1, v_2, \dots, v_{|V|} \rangle$. For any $0 \leqslant t \leqslant |V|$, let $V_t = \langle v_1, v_2, \dots, v_t \rangle$ be the set of the first t nodes in the order σ. We refer to these nodes as the nodes that have *arrived until time* t, and we refer to v_t as the node arriving at time t. We let G_t be the signed subgraph of G induced by V_t with the sign induced by s on the edges whose both endpoints are in V_t. We also denote by $N_{G_t}(u)$ the set of neighbors of node u in G_t that share a positive edge with u under the sign function s and by f_t^{OPT} an assignment function which induces an optimal correlation clustering solution for graph G_t. Note that the solution of an algorithm Alg on an online instance $\mathcal{I}$ can be described as a sequence of assignment functions $f_1, f_2, \dots, f_{|V|}$, denoted for simplicity by $f_{1,2,\dots,|V|}$. We say that $f_{1,2,\dots,|V|}$ is a c-approximation to the optimal solution if $\text{cost}(f_t) \leqslant c \cdot \text{cost}(f_t^{OPT})$, $\forall t \in \{1, 2, \dots, |V|\}$. Additionally, an algorithm is a c-competitive algorithm if the solution it produces is a c-approximation for all instances $\mathcal{I}$.

The recourse of a node u, $r(u)$, that arrived at time t is the number of times the assignment function sequence changes the cluster id assigned to u. That is $r(u) = \sum_{t' > t} \mathbb{1}\{f_{t'-1}(u) \neq f_{t'}(u)\}$. The recourse of an algorithm is the worst case recourse over all instances $\mathcal{I}$ and nodes u.

Algorithm 1 AGREEMENT

```
1: Input: A signed graph $G'$, and a parameter $\epsilon$
2: Output: A clustering $\mathcal{C}$ of $G'$
3: Initialize: $G \leftarrow E^+(G'), \hat{G} \leftarrow E^+(G')$
4: Step 1: Remove all edges of $\hat{G}$ whose endpoints are not in $\epsilon$-agreement in $G$.
5: Step 2: Remove all edges of $\hat{G}$ whose endpoints are both light.
6: Step 3: Compute the connected components $\tilde{\mathcal{C}}$ of $\hat{G}$.
7: return $\tilde{\mathcal{C}}$
```

3. Algorithm

In this section we present our algorithm along with a proof sketch of the guarantees that it provides (the full proofs are available in the appendix). Before we describe the algorithm, we need to introduce some notation. Let $\mathcal{C}_t$ be the clustering of G_t induced by the assignment function f_t. A cluster is called singleton if it contains a single node. We denote by $S(\mathcal{C}_t)$ the set of nodes that belong to a singleton cluster in $\mathcal{C}_t$. Respectively, we use $H(\mathcal{C}_t)$ to denote the set of nodes that belong to a non-singleton cluster in $\mathcal{C}_t$.

The Agreement algorithm. Our algorithm uses as a subroutine the static algorithm introduced in (Cohen-Addad et al., 2021); we refer to this algorithm as the *Agreement* algorithm. The Agreement algorithm makes use of the following definitions and lemmas. Given a signed graph G_t, the Agreement algorithm is executed on the graph $G = (V(G_t), E^+(G_t))$. In the following, $N_G(u)$ denotes the neighborhood of node u in G.

Definition 1. *Two nodes u, v are in ϵ-agreement in G if $|N_G(u) \triangle N_G(v)| < \epsilon \max\{|N_G(u)|, |N_G(v)|\}$, where the symbol $\triangle$ denotes the symmetric difference of two sets.*

Definition 2. *A node u is light if it is in ϵ-agreement with less than an ϵ fraction of its neighborhood.*

Since the Agreement algorithm is a central ingredient of our algorithm, and our analysis depends on its properties, we present it in Algorithm 1.

At a high-level the algorithm performs lazy cluster-assignment updates. Namely, at each time t, our algorithm re-runs the Agreement algorithm (which produces an $O(1)$-approximate solution for graph G_t) and only updates the cluster ids of the vertices that joined a different (non-singleton) cluster at time t compared to the clustering at time $t-1$. A final rule changes the ids of clusters that have grown in size by a constant factor. We use $\widetilde{\mathcal{C}}_t$ to refer to the clustering produced by running the Agreement algorithm on G_t, and $\tilde{f}_t$ to refer to the assignment function that induces $\widetilde{\mathcal{C}}_t$. Similarly, we use $\tilde{f}_t(C)$ to refer to the unique cluster id of a cluster $C \in \widetilde{\mathcal{C}}_t$.

Evolving clusters. We keep track of clusters that evolve over time. Each evolving cluster is associated with a unique cluster id. The life-cycle of a cluster starts whenever a cluster id is seen for the first time, and it ends whenever no node uses the cluster id anymore. We monitor the growth of each evolving cluster using the following definition.

Definition 3 (Origin cluster). *Let ID be a cluster id that is used by the assignment function f_t, and let t_{min} be the minimum t for which ID is used by $f_{t_{min}}$. The origin cluster of ID, denoted by $Origin(ID)$, is the cluster with id ID in the clustering induced by $f_{t_{min}}$.*

Once a cluster $C \in \mathcal{C}_t$ becomes a constant factor larger in size compared to the origin cluster with the same id, we assign a fresh id to C (hence making C the origin cluster of that fresh id). We do this so that we can later bound the competitive ratio of the clusters that we output.

The algorithm. Our *Online Agreement (Agree-On)* algorithm is described in details in Algorithm 2. At a high-level, the algorithm has three main phases which are executed for every node-arrival.

- *Offline re-clustering phase:* We run the Agreement algorithm on G_t. Let $\widetilde{\mathcal{C}}_t$ be the resulting clustering and $\tilde{f}_t$ be the assignment function inducing $\widetilde{\mathcal{C}}_t$.
- *Initial assignment phase:* This phase combines $\widetilde{\mathcal{C}}_{t-1}, \widetilde{\mathcal{C}}_t$ as well as the previously maintained clustering $\mathcal{C}_{t-1}$ to produce $\mathcal{C}_t$. First, it assigns a new unique id to the newly arrived node v_t, if $\{v_t\}$ is a singleton cluster of $\widetilde{\mathcal{C}}_t$. Then it applies the following three rules to set the ids of all nodes (except possibly v_t):
  - **Rule 1**: For each non-singleton cluster $C \in \widetilde{\mathcal{C}}_t$ if there exists a non-singleton intersecting cluster $C' \in \widetilde{\mathcal{C}}_{t-1}$ then assign the cluster id of $f_{t-1}(C')$ to all nodes in C. This essentially identifies C and C' to be part of the same evolving cluster. We exploit structural properties to show that any non-singleton cluster $C \in \widetilde{\mathcal{C}}_t$ intersects at most one non-singleton cluster $C' \in \widetilde{\mathcal{C}}_{t-1}$.
  - **Rule 2**: For each non-singleton cluster $C \in \widetilde{\mathcal{C}}_t$ that consists entirely of nodes that were in singleton clusters of $\widetilde{\mathcal{C}}_{t-1}$ (but not necessarily in singleton clusters of $\mathcal{C}_{t-1}$), and potentially the newly arrived node, do the following. If there is a cluster $C' \in \mathcal{C}_{t-1}$ containing the majority of the nodes in C, then C gets the same cluster id as C'; otherwise C receives a new unique cluster id and C becomes the origin cluster of that id.
  - **Rule 3**: ensures that each singleton cluster in $\widetilde{\mathcal{C}}_t$ retains its previous cluster id.

  At the end of this phase, for each non-singleton cluster $C \in \widetilde{\mathcal{C}}_t$ we have assigned the id of a cluster $C' \in \mathcal{C}_t$

to all nodes in C. This assignment induces a mapping from ids of non-singleton clusters $C \in \widetilde{\mathcal{C}}_t$ to ids of non-singleton clusters $C' \in \mathcal{C}_t$; we represent this mapping using the assignment function ϕ_t.

- *Assignment refinement phase:* After forming an initial clustering of $\mathcal{C}_t$, this phase identifies clusters $C \in \widetilde{\mathcal{C}}_t$ that are significantly larger compared to the size of their origin cluster of $\phi_{\tilde{f}_C}$. Such clusters become new origin clusters and receive a new unique cluster id.

Algorithm intuition: The crux of our algorithm is the combination of the notion of origin clusters together with $\tilde{f}_t$, $\tilde{f}_{t-1}$, and f_{t-1}. Before presenting some useful structural properties of our algorithm, we show that the algorithm is well defined; that is, we show that for any time t, $\mathcal{C}_t$ indeed forms a partition of V_t. We start from the fact that $\widetilde{\mathcal{C}}_t$ is a partition of V_t. Each non-singleton cluster of $C \in \widetilde{\mathcal{C}}_t$ is assigned an id by either Rule 1 or Rule 2, since C consists of nodes that form singleton clusters of $\widetilde{\mathcal{C}}_{t-1}$ and at most one non-singleton cluster $C' \in \widetilde{\mathcal{C}}_{t-1}$ intersecting C. All singleton clusters of $\widetilde{\mathcal{C}}_t$ are assigned an id by Rule 3; if v_t is a singleton cluster in $\widetilde{\mathcal{C}}_t$, then it is assigned an id right before the main loop of the algorithm.

In the appendix we prove several properties of our algorithm with respect to the assignment functions $\tilde{f}_t, f_t$. We mention these properties and refer to the appendix for their proof.

1. Let C be a non-trivial cluster of $\widetilde{\mathcal{C}}_t$. Then, at time t all nodes of C are assigned the same cluster id in f_t. This is ensured by **Rule 1** and **Rule 2**.

2. Let C, C' be two non-trivial clusters of $\widetilde{\mathcal{C}}_t$. Then, f_t assigns to all nodes in cluster C a cluster id that is distinct from the one assigned to the nodes in C'.

3. Let C be a non-trivial cluster of $\widetilde{\mathcal{C}}_t$, we denote by $S_t^C = \{v \in S(\widetilde{\mathcal{C}}_t) : f_t(v) = f_t(C)\}$ the set of nodes which do not belong to C but are clustered together with C in $\mathcal{C}_t$. Note that: (a) for each $u \in S_t^C$ it holds $f_t(u) \neq \tilde{f}_t(u)$; and (b) every node u for which it holds $f_t(u) = f_t(C)$ belongs to $C \cup S_t^C$.

To simplify our proof sketch we make the following simplifying assumption regarding the structure of $\widetilde{\mathcal{C}}_t$. (Please refer to the appendix for the full proofs not using these assumption).

Assumption 3.1. Let $C \in \widetilde{\mathcal{C}}_t$ be a non-trivial cluster and let $u \in C$. Node u has, in G_t, at least $(1-\epsilon)|C|$ positive edges to nodes in C and at most $\epsilon|C|$ positive edges to nodes outside C.

Algorithm 2 Online Agreement (Agree-On)

```
1:  Input: An online instance $\mathcal{I} = (G, \sigma)$, a parameter $\epsilon$
2:  Output: An assignment function sequence $f_{1,2,\dots,|V|}$
3:  Initialization: $G_{-1} \leftarrow \emptyset$, $ID_{next} \leftarrow 0$
4:  on arrival of $v_t$ do
5:    $G_t \leftarrow G_{t-1} \cup \{v_t\}$
6:    $\widetilde{\mathcal{C}}_t \leftarrow$ Agreement$(G_t, \epsilon)$
7:    if $v_t \in S(\widetilde{\mathcal{C}}_t)$ then
8:      $Origin(ID_{next}) \leftarrow \{v_t\}$
9:      $ID_{next} \leftarrow ID_{next} + 1$
10:   end if
11:   for all $C \in \widetilde{\mathcal{C}}_t$ s.t. $|C| > 1$ do
12:     //* Rule 1:
13:     if $\exists\, C' \in \widetilde{\mathcal{C}}_{t-1}$ s.t. $C \cap C' \neq \emptyset, |C'| > 1$ then
14:       for $u \in C$ do $f_t(u) \leftarrow f_{t-1}(C')$ endfor
15:     end if
16:     //* Rule 2:
17:     if $C \setminus v_t \subseteq S(\widetilde{\mathcal{C}}_{t-1})$ then
18:       if $\exists C' \in \mathcal{C}_{t-1}$ s.t. $|C' \cap C| > |C|/2$ then
19:         for $u \in C$ do $f_t(u) \leftarrow f_{t-1}(C')$ endfor
20:       else
21:         for $u \in C$ do $f_t(u) \leftarrow ID_{next}$ endfor
22:         $Origin(ID_{next}) \leftarrow C$
23:         $ID_{next} \leftarrow ID_{next} + 1$
24:       end if
25:     end if
26:   end for
27:   //* Rule 3:
28:   for all $u \in S(\widetilde{\mathcal{C}}_t) \setminus v_t$ do $f_t(u) \leftarrow f_{t-1}(u)$ endfor
29:   //* Store $\phi_t$
30:   for all $C \in \widetilde{\mathcal{C}}_t$ s.t. $|C| > 1$ do
31:     $\phi_t(\tilde{f}_t(C)) \leftarrow f_t(C)$
32:   end for
33:   //* Assignment refinement phase
34:   Let $\mathcal{C}'$ be the current clustering induced by $f_t$.
35:   for all $C \in \widetilde{\mathcal{C}}_t$ s.t. $|C| > 1$ do
36:     Let $C' \in \mathcal{C}'$ be the cluster s.t. $C \subseteq C'$.
37:     if $|C| \geqslant (3/2)|Origin(f_t(C'))|$ then
38:       for $u \in S(\widetilde{\mathcal{C}}_t)$ do $f_t(u) \leftarrow ID_{next}$ endfor
39:       $ID_{next} \leftarrow ID_{next} + 1$
40:       $Origin(ID_{next}) \leftarrow C$
41:     end if
42:   end for
43: end on
```

3.1. Bounding the competitive ratio

In this section we give an overview on how we prove that $\mathrm{cost}(f_t) \leqslant \Theta(\mathrm{cost}(f_t^{OPT}))$. We do this by proving $\mathrm{cost}(f_t) \leqslant \Theta(\mathrm{cost}(\tilde{f}_t))$, which combined with the fact that $\mathrm{cost}(\tilde{f}_t) \leqslant \Theta(\mathrm{cost}(f_t^{OPT}))$ implies the constant competitive ratio of our algorithm.

For convenience, we use $E^+(X, Y)$ (resp., $E^-(X, Y)$) to denote the edges in $E^+(G_t) \cap (X \times Y)$ (resp., $E^-(G_t) \cap (X \times Y)$).

The proof consists of two parts. First, we prove that the cost incurred on f_t by the edges on nodes in $S(\tilde{f}_t)$ is bounded by $\Theta(\text{cost}(f_t^{OPT}))$. Then, we move to showing that the cost incurred on f_t by the edges on nodes in $H(\tilde{f}_t)$ is also $\Theta(\text{cost}(f_t^{OPT}))$.

We first sketch the first part of the proof. Notice that the cost incurred on f_t for the edges of a node $u \in S(\tilde{f}_t)$ is only larger than the cost incurred on $\tilde{f}_t$ for the edges of u if f_t clusters u in a non-singleton cluster C. Fix a cluster $C \in \mathcal{C}_t$. We bound $|E^-(C, C)|$ by $\Theta(|E^+(S(\tilde{f}_t) \cap C, V_t)|)$. To do so, we prove that C is not much larger than its origin cluster and that further it holds that $|E^-(u, C)| \geqslant \Theta(|C|)$ for each $u \in S(\tilde{f}_t) \cap C$. The latter is proved by using properties of the clusters generated by the Agreement algorithm.

Now we move to sketching the second part of the proof. Let $C_1, C_2, \ldots, C_k$ be the set of non-singleton clusters of $\widetilde{\mathcal{C}}_t$, then f_t and $\tilde{f}_t$ disagree exactly on $S_t^{C_1}, S_t^{C_2}, \ldots, S_t^{C_k}$. Fix a specific pair C, S_t^C, and note that for this pair:

1. $\tilde{f}_t$ pays $|E^+(C \cup S_t^C, V_t \setminus C \cup S_t^C)| + |E^+(C, S_t^C)| + |E^-(C, C)|$.
2. f_t pays $|E^+(C \cup S_t^C, V_t \setminus C \cup S_t^C)| + |E^-(C, S_t^C)| + |E^-(S_t^C, S_t^C)|$

Combining this, with the first part of the proof, we get that $\text{cost}(f_t)$ is at most:

$$\left(\sum_{C \in \widetilde{\mathcal{C}}_t : |C| > 1} |E^-(C, S_t^C)| + |E^-(S_t^C, S_t^C)| \right) + \Theta(\text{cost}(\tilde{f}_t))$$

$$\leqslant \left(\sum_{C \in \widetilde{\mathcal{C}}_t : |C| > 1} |S_t^C||C| + |S_t^C|^2 \right) + \Theta(\text{cost}(\tilde{f}_t))$$

and,

$$\text{cost}(\tilde{f}_t) \geqslant \sum_{C \in \widetilde{\mathcal{C}}_t : |C| > 1} |E^+(C, S_t^C)|$$

To conclude, we are left to argue that $|S_t^C||C| + |S_t^C|^2 = \Theta(|E^+(C, S_t^C)|)$ for any non-trivial cluster $C \in \widetilde{\mathcal{C}}_t$.

Let $C \in \widetilde{\mathcal{C}}_t$ be a cluster whose nodes get assigned cluster id after $\tilde{f}_t(C)$.

Note that the assignment refinement phase, takes an initial version of $\mathcal{C}_t$ and if there is a cluster $C \in \widetilde{\mathcal{C}}_t$ that is significantly larger than $Origin(f_t(C))$, then all nodes in C receive a new cluster id in $\mathcal{C}_t$ and the origin cluster of the newly formed cluster is set to be C. At a high level the assignment refinement phase, forces a cluster of f_t to remain as similar as possible to its origin cluster. For simplicity here we make the following assumption (for a full proof without this assumption please refer to the appendix).

Assumption 3.2. Let $C' \in \widetilde{\mathcal{C}}_{t'}$ and $C \in \widetilde{\mathcal{C}}_t$ be two non-singleton clusters, for $t' < t$. If they are part of the same evolving cluster (i.e., if $\phi_{t'}(\tilde{f}_{t'}(C')) = \phi_t(\tilde{f}_t(C))$), then $|C' \cap C''| \geqslant (1/2) \max\{|C'|, |C''|\}$.

We start by lower bounding the number of positive edges between nodes in S_t^C and nodes in C, i.e., $|E^+(C, S_t^C)|$. Note that any node $u \in S_t^C$ must have been part of a non-trivial cluster $C' \in \widetilde{\mathcal{C}}_{t'}$ s.t. $\phi_{t'}(\tilde{f}_{t'}(C')) = \phi_t(\tilde{f}_t(C))$, for some time $t' < t$. Hence, from Assumption 3.2 we get that $|C \cap C'| \geqslant (1/2) \max\{|C|, |C'|\}$. Moreover, from Assumption 3.1 we know that node u has at least $(1 - \epsilon)|C'|$ positive edges inside C'. Combining the latter two facts we can conclude that node u has at least $(1/2 - \epsilon)|C|$ edges shared with nodes in C. Thus, $|E^+(C, S_t^C)| \geqslant |S_t^C|(1/2 - \epsilon)|C|$. Combining the latter with the trivial upper bound $|E^+(C, S_t^C)| \leqslant |S_t^C||C|$ we get that $|E^+(C, S_t^C)| = \Theta(1)|S_t^C||C|$.

To conclude the proof it suffices to show that $|S_t^C| = \Theta(1)|C|$. From assumption 3.1 we have that positive edges between nodes in C and nodes outside C are at most $\epsilon|C|^2$. On the other hand, we just argued that $|E^+(C, S_t^C)| \leqslant \Theta(1)|S_t^C||C|$. By combining these two facts we get that $\Theta(1)|S_t^C||C| < \epsilon|C|^2$ and consequently $|S_t^C| \leqslant \Theta(1)|C|$.

3.2. Bounding the recourse

At a high level, the recourse is at most $O(\log|V|)$ because, roughly speaking, whenever a node changes cluster id then the size of the origin cluster corresponding to the new cluster id is a multiplicative factor larger than the size of the origin cluster which corresponds to the old cluster id. The proof of the latter is quite technical and we defer it to section F of the appendix.

4. Lower Bound

In this section we present an online correlation clustering instance where any algorithm that achieves constant competitive ratio requires $\Omega(\log n)$ recourse per node, in the worst case, where n the total number of nodes in the end.

Let $c \geqslant 1$ be a constant. Initially our instance reveals two nodes u, v with a positive edge between them. The rest of the node arrival sequence works in logarithmically many phases. Our instance forces any c-competitive algorithm $\mathcal{A}$ to behave as follows. At odd phases $\mathcal{A}$ clusters u, v together while at even phases it clusters u, v separately. Let r be the total number of phases, then note that such a construction forces either u or v to have a recourse of at least $r/4$.

Lemma 4.1. *Let $c \geqslant 1$ be a constant. Any online algorithm for the correlation clustering problem with a competitive ratio smaller than c has a recourse $\Omega(\log(n))$ where n denotes the total number of nodes.*

Proof. For a simpler description, we are using an unsigned graph to describe our construction. The actual signed instance is constructed by replacing every existing edge with a '+' edge, while every missing edge between two already revealed nodes with a '-' edge.

We use two main gadgets: 1) cliques $C^s_{u,v}$ of size s whose nodes are connected to both u and v, and 2) cliques C^s_u (resp., C^s_v) of size s whose nodes are all connected to u (resp., v). We note that different cliques share no edge between them.

Initially, our adversarial node arrival sequence reveals the two base nodes. At that point any algorithm with a bounded competitive ratio connects the two nodes since the optimum solution has cost 0.

In phase 0, our adversarial sequence reveals a pair of cliques C^{2c}_u, C^{2c}_v. Note that the optimal solution cluster u with C^{2c}_u in a single cluster and v with C^{2c}_v in a second cluster, achieving a total cost of 1 (the positive edge between the base nodes). Any solution that clusters u, v together costs at least $2c$ as any splitting of the two clusters leaves at least $2c - 1$ inter-clusters edges while merging the two clusters costs $\Omega(c^2)$. Thus, any algorithm with a competitive ratio less than c splits the nodes u, v to two clusters.

In phase 1, our sequence reveals a clique $C^{(2c)^3}_{u,v}$. The optimal solution costs $OPT = 2 \cdot 2c$ and is achieved by clustering $u, v, C^{(2c)^3}_{u,v}$ in the same cluster and each other clique into a separate cluster. Notice that any splitting of the cluster $C^{(2c)^3}_{u,v}$ leaves at least $(2c)^3 - 1$ inter-clusters edges. Hence, any c-competitive online algorithm clusters u, v together with clique $C^{(2c)^3}_{u,v}$, otherwise it costs at least $(2c)^3 - 1 > c \cdot 4c = c \cdot OPT$.

Following the same pattern, at even phases i our sequence reveals the pair of cliques $C^{(2c)^{3i}}_u$ and $C^{(2c)^{3i}}_v$ forcing any c-competitive algorithm to cluster u and v separately, while at odd phases i our sequence reveals the clique $C^{(2c)^{3i}}_{u,v}$ forcing any c-competitive algorithm to cluster u and v together.

We next prove the claim for even and odd phases separately. First, assume i is even. At the end of phase i, the optimum solution clusters v with $C^{(2c)^{3i}}_v$ in a single cluster, u with $C^{(2c)^{3i}}_u$ into a second cluster, and every other clique into a separate cluster on its own. The above clustering has a cost of $OPT = 2 \cdot \left(((2c)^3 + (2c)^6 + \cdots + (2c)^{3(i-1)}\right)$. Any algorithm where u, v are in the same cluster has a cost of $(2c)^{3i} > c \cdot 2 \cdot \left(((2c)^3 + (2c)^6 + \cdots + (2c)^{3(i-1)}\right) \geqslant c \cdot OPT$. Thus, any c-competitive online algorithm assign u and v to different clusters.

Next, assume i is odd. At the end of phase i, the optimum solution is formed by clustering u, v, and $C^{(2c)^{3i}}_{u,v}$ together and every other clique into a separate cluster on its own. The above clustering has a cost of $OPT = 2 \cdot \left(((2c)^3 + (2c)^6 + \cdots + (2c)^{3(i-1)}\right)$. Any algorithm where u, v are in different clusters has a cost of $(2c)^{3i} > c \cdot 2 \cdot \left(((2c)^3 + (2c)^6 + \cdots + (2c)^{3(i-1)}\right) \geqslant c \cdot OPT$. Thus, any c-competitive online algorithm clusters u and v together.

Let r be the number of phases . Given that every 2 phases, u, v are clustered together and then separately again, at least one of u, v needs to increase its recourse by at least 1. Thus, after r rounds, one of u, v incurs recourse at least $r/4$.

To end the proof let the last phase r be an odd round, then the total number of nodes is $n = 2 + 2 \cdot 2c + (2c)^3 + 2 \cdot (2c)^6 + (2c)^9 + ... + (2c)^{3r} < 2 \cdot (2c)^{3r+1} < (2c)^{3r+2}$. Thus, $r = \Omega(log(n)/log(c))$ which concludes the proof. □

5. Experiments

5.1. Datasets

Our study includes four graphs that are formed by user-to-user interactions. Specifically, we consider a Social network (musae-facebook), an email network (email-Enron), a collaboration network (ca-AstroPh), and a paper citation network (cit-HepTh). All but the cit-HepTh datasets are static undirected graphs. As an effort to considered a realistic dataset for experimenting with online algorithms, we use the dataset cit-HepTh which has timestamps on the nodes indicating a natural arrival order. Since cit-HepTh is directed, we transform it to an undirected graph by ignoring edge directions. In all of our datasets we removed all parallel-edges. Our datasets are obtained from SNAP (Leskovec & Krevl, 2014); their basic characteristics are summarized in Table 1.

	#nodes	#edges	online?
musae-facebook	22,470	171,002	no
email-Enron	36,692	183,831	no
ca-AstroPh	18,772	198,110	no
cit-HepTh	27,770	352,807	yes

Table 1. Basic characteristics of our datasets.

5.2. Baselines

PIVOT. As one of the baselines, we consider the 3-approximate pivoting algorithm introduced by Ailon et al. (Ailon et al., 2008). We refer to this algorithm as PIVOT. In the offline model, PIVOT works as follows. First, it creates a random order of the nodes, marks all nodes as unclustered, and then iterates over the nodes in the aforementioned random order. If the current node v is still unclustered, then v forms a cluster together with all of its unclustered neighbors and they are marked as clustered.

The naive way to use PIVOT in the online setting is to re-run it from scratch following every node arrival, each time with a fresh random order. However, we re-use the same random order of the previously arrived nodes, which leads to an improved practical performance both in terms of recourse and running time. That is, whenever a new node arrives it is inserted in the preexisting random order at a random position (the relative order of the previously arrived nodes remains unchanged). We call the latter order the *new random order*. It is not hard to verify that all clustering choices of the algorithm regarding nodes which precede the newly arrived node in the new random order remain the same compared to the execution of the algorithm in the previous iterations (before the arrival of the last node). Thus, we only simulate the algorithm starting from visiting the index of the newly arrived node in the new random order onward.

AGREE-OFF. This baseline reruns the Agreement algorithm following each node arrival. In our experiments we set the parameters $\beta = \lambda = 0.2$, as this setting exhibited the best behavior in (Cohen-Addad et al., 2021).

Greedy recourse minimization. For the baselines we assume that there is no consistent cluster id assignment to the clusters produced after each node arrival, and hence, measuring recourse as per our definition would give an unfair advantage to our (more designated) algorithm. To have a fair comparison, we measure recourse independently of the actual cluster ids that each algorithm assigns. That is, given two clusterings $\mathcal{C}_{t-1}, \mathcal{C}_t$ from consecutive runs of some algorithm we reassign the cluster ids of $\mathcal{C}_t$ such that we minimize the number of nodes that have distinct cluster ids in $\mathcal{C}_{t-1}$ and $\mathcal{C}_t$. Specifically, we construct a bipartite graph $B = (U \cup V, E)$ where each node of U (resp., V) represents a cluster $C_u \in \mathcal{C}_{t-1}$ (resp., $C_v \in \mathcal{C}_t$), and there is an edge $(u, v) \in E \cap \{U \times V\}$ with weight w if C_u overlaps with C_v on w nodes (if the C_u and C_v do not overlap, there is no edge (u, v)). To re-assign the ids of the clusters in $\mathcal{C}_t$, we run the greedy 2-approximate maximum bipartite matching on B, and each cluster $C \in \mathcal{C}_t$ that is matched with a cluster $C' \in \mathcal{C}_t$ gets the same cluster id as C; if C' remains unmatched then it receives a new unique cluster id [2]. Finally, the recourse is computed by counting the number of nodes with distinct cluster ids in $\mathcal{C}_{t-1}$ and $\mathcal{C}_t$. We apply this post-processing to all algorithms (including ours).

[2]Although one can solve the maximum bipartite matching exactly, this task is expensive and orthogonal to our study.

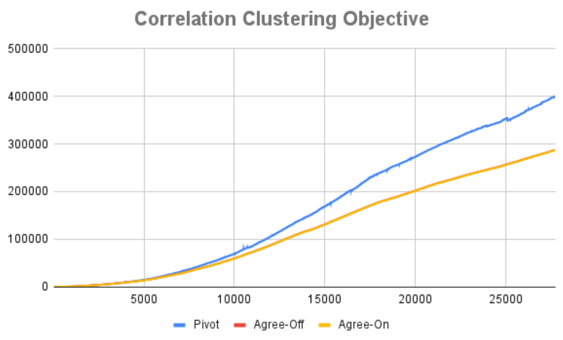


Figure 1. The cost of the clustering produced by the different algorithms for the dataset cit-HepTh.

5.3. Setup

Our code is written in C++ and is available online[3]. We run our experiments on a e2-standard-16 Google Cloud instance, with 16 cores, 2.20GHz Intel(R) Xeon(R) processor, and 64 GiB main memory.

For the datasets email-Enron, ca-AstroPh, musae-facebook, we use a random arrival order, as the nodes are not timestamped. For the dataset cit-HepTh, we consider the arrival order implied by the timestamps on the nodes. Once a node v arrives, it reveals only its edges to the previously arrived nodes (the remaining edges are revealed once the other endpoint of each such edge arrives).

5.4. Results

Solution quality. Figure 1 shows the cost of the cluster produced by each of the algorithms that we consider throughout the sequence of node arrivals, on the dataset cit-HepTh. The plots for the other datasets are shown in Figure 5 in the appendix. In all datasets but ca-AstroPh the AGREE-OFF and AGREE-ON perform significantly better than PIVOT. The latter implies that the solution calculated by AGREE-ON is at most a 3-approximation of the offline optimum. We would like to underline that the best algorithm in terms of approximation guarantees for the correlation clustering problem is an LP-based variation of PIVOT in (Chawla et al., 2015). However, the latter algorithm due to its high computational cost becomes quickly infeasible to run even for graphs where the number of nodes is one or two orders of magnitude smaller than the ones we use in our experiments. Thus, the state-of-the-art algorithm for the correlation clustering problem both in terms of practical performance and provable guarantees is PIVOT, and we successfully compare against it.

While not immediately visible from the plots, AGREE-ON performs slightly better than AGREE-OFF in all datasets.

[3]`https://github.com/google-research/google-research/tree/master/online_correlation_clustering`

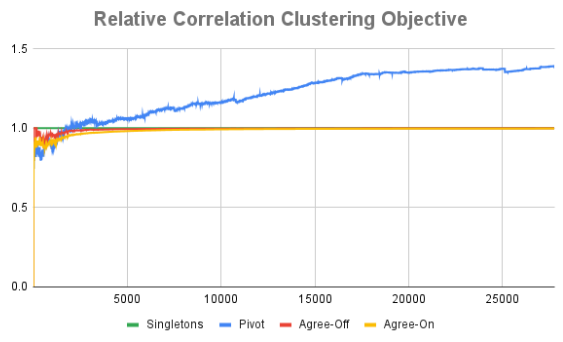


Figure 2. The quality of the clustering produced by the different algorithms for the dataset cit-HepTh, relatively to SINGLETONS.

	PIVOT	AGREE-OFF	AGREE-ON
email-Enron	14	3	1
ca-AstroPh	10	2	1
musae-facebook	10	3	1
cit-HepTh	13	5	1

Table 2. The maximum recourse per node over the whole sequence of node arrivals, for the four datasets that we considered.

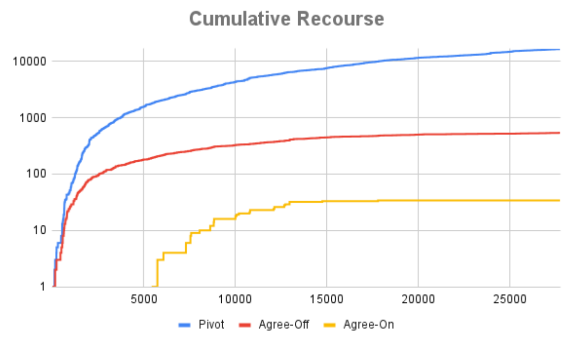


Figure 3. The total recourse incurred by the algorithms that we consider on the dataset cit-HepTh, in log scale.

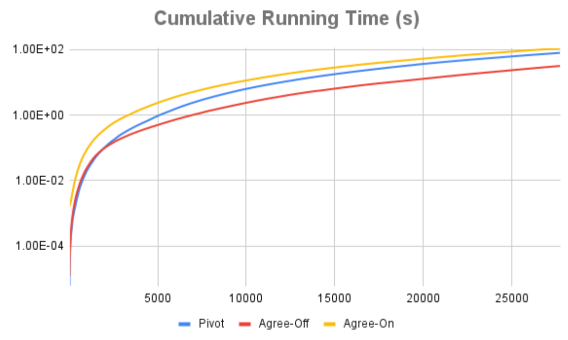


Figure 4. The running time (in seconds) of the algorithms that we consider on the dataset cit-HepTh, in log scale.

This is due to the fact that once a set of nodes C is clustered together at time t_1, AGREE-ON keeps the nodes in C clustered together at times $t > t_1$ even if the nodes in C obtain many outgoing edges from C, whereas AGREE-OFF would split C under such a scenario.

Our datasets are sparse throughout the arrival sequence. Sparse graphs tend to not have a good correlation clustering structure, and often the clustering that consists of only singleton clusters is a competitive solution; we denote such a solution as SINGLETONS. We illustrate this in Figure 2 for the dataset cit-HepTh (and in Figure 6, in the appendix, for the other datasets), where we present the cost function relatively to SINGLETONS. After a small number of node arrivals, PIVOT performs clearly worse than SINGLETONS, while AGREE-OFF and AGREE-ON always perform better or equally well compare to SINGLETONS. These findings are not surprising. On the one hand, PIVOT is more likely to create clusters with very small density which results to a lot of negative intra-cluster edges. On the other hand, AGREE-OFF (and hence AGREE-ON) creates singleton clusters if the graph contains no clear clustering structure; which implies that AGREE-OFF produces a clustering that is never worse than the one produced by SINGLETONS. Unlike the case of our particular datasets, AGREE-OFF (and consequently AGREE-ON) often produces many medium-sized clusters on sparse graphs[4] (Cohen-Addad et al., 2021) implying performance significantly better that SINGLETONS.

[4]Unfortunately, it was not reasonable to run the whole arrival sequence on the same datasets, due to their size.

Recourse. Table 2 shows the maximum recourse of a node, for each dataset. Both AGREE-OFF and AGREE-ON perform significantly better than PIVOT, while AGREE-ON never changes the cluster id of a node twice. Recall that the greedy reassignment post-processing benefits all algorithms.

Next, we investigate the total recourse that each algorithm incurs over the whole node-arrival sequence, i.e., the sum of the recourse of each node. The plot for the dataset cit-HepTh is shown in Figure 3. The plots for the other datasets are in Figure 5 in the appendix. AGREE-ON consistently incurs 1-2 order of magnitude less recourse compared to PIVOT, and up to an order of magnitude less recourse compared to AGREE-OFF.

Running time. For simplicity the version of AGREE-ON described in 2 computes the clustering $\tilde{C}_t$ by naively rerunning the Agreement algorithm 1 on the graph G_t. However, we can easily use most of the prior information to compute the clustering $\tilde{C}_t$ without rerunning the Agreement algorithm from scratch. To this end, note that for any node $u \in G_t \backslash \{v_t\}$ its degree can be updated in $O(1)$ by checking if u and the newly arrived node v_t share an edge. Moreover let $u, v \in G_{t-1}$ be two nodes which share an edge, then the size of the symmetric difference of their neighborhoods can be updated in $O(1)$ by checking if the newly arrived node is the neighbor of exactly one of them. Thus, all $\epsilon-$agreements between nodes $u, v \in G_{t-1}$ can be updated in linear time. In addition, checking the $\epsilon-$agreement between two nodes u, v without any prior information takes time $O(\min\{d_u, d_v\})$. Consequently we can check the $\epsilon-$agreement between the newly arrived node and all its neighbors in time $O(d_{v_t}^2)$. Finally, lines $7 - 43$ of AGREE-ON for each new node arrival can be easily implemented in linear time. Overall, a careful implementation of AGREE-ON would yield to an algorithm whose complexity for the whole arrival sequence is $O(\sum_u d_u^2) + n \cdot O(n+m) = O(nm)$ (matching Pivot's complexity) instead of $O(n^2 m)$ from a naive implementation of the pseudocode.

As expected, in terms of running time, AGREE-OFF performs comparably to PIVOT as shown in Figure 4 for the dataset cit-HepTh (and in Figure 5, in the appendix, for the other datasets). On the other hand, due to the required post-processing (lines $7 - 43$ of the pseudocode) following the computation made by AGREE-OFF, AGREE-ON performs less than $2\times$ slower compared to AGREE-OFF and PIVOT. Since the latter post-processing requires linear time, this seems to imply that in practice all these three algorithms require $O(n)$ time per node arrival. This is in contrast to their trivial worst-case performance of $O(\sum_{v\in V} d^2(v))$ time for AGREE-OFF and $O(m)$ time for PIVOT, per update.

Take away message. Cluster stability is an important property of online and dynamic algorithms. Our experiments suggest that AGREE-ON not only significantly outperforms the baselines in terms of recourse, but produces solutions comparable and often better compared to the state-of-the-art algorithms.

Conclusions and Future Work

We present a new algorithm for online correlation clustering with strong consistency guarantees. In particular, our algorithm is simple, efficient and obtains optimal recourse. We also study the empirical performance of our algorithm showing its effectiveness in practice.

A natural future direction is to improve the approximation factor obtained by our algorithm and to study other classic clustering problems in the online setting with recourse.

6. Acknowledgments

Andreas Maggiori was supported by the Swiss National Science Fund (SNSF) grant no 200020–182517 "Spatial Coupling and Interpolation Methods for the Analysis of Coding and Estimation". Part of this work was done while Andreas Maggiori was an intern at Google Research, hosted by Nikos Parotsidis.

References

Agrawal, R., Halverson, A., Kenthapadi, K., Mishra, N., and Tsaparas, P. Generating labels from clicks. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pp. 172–181, 2009.

Ailon, N., Charikar, M., and Newman, A. Aggregating inconsistent information: Ranking and clustering. *J. ACM*, 55(5), nov 2008. ISSN 0004-5411. doi: 10.1145/1411509.1411513. URL https://doi.org/10.1145/1411509.1411513.

Arasu, A., Ré, C., and Suciu, D. Large-scale deduplication with constraints using dedupalog. In *2009 IEEE 25th International Conference on Data Engineering*, pp. 952–963. IEEE, 2009.

Bansal, N., Blum, A., and Chawla, S. Correlation clustering. *Machine learning*, 56(1):89–113, 2004.

Bonchi, F., Gionis, A., and Ukkonen, A. Overlapping correlation clustering. *Knowledge and information systems*, 35(1):1–32, 2013.

Chakrabarti, D., Kumar, R., and Punera, K. A graph-theoretic approach to webpage segmentation. In *Proceedings of the 17th international conference on World Wide Web*, pp. 377–386, 2008.

Chawla, S., Makarychev, K., Schramm, T., and Yaroslavtsev, G. Near optimal lp rounding algorithm for correlation-clustering on complete and complete k-partite graphs. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pp. 219–228, 2015.

Chen, Y., Sanghavi, S., and Xu, H. Clustering sparse graphs. In *Proceedings of the 25th International Conference on Neural Information Processing Systems-Volume 2*, pp. 2204–2212, 2012.

Cohen-Addad, V., Hjuler, N., Parotsidis, N., Saulpic, D., and Schwiegelshohn, C. Fully dynamic consistent facility location. In *NeurIPS'19-33rd Conference on Neural Information Processing Systems*, 2019.

Cohen-Addad, V., Lattanzi, S., Mitrovic, S., Norouzi-Fard, A., Parotsidis, N., and Tarnawski, J. Correlation clustering in constant many parallel rounds. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 2069–2078. PMLR, 2021.

Demaine, E. D., Emanuel, D., Fiat, A., and Immorlica, N. Correlation clustering in general weighted graphs. *Theoretical Computer Science*, 361(2-3):172–187, 2006.

Fichtenberger, H., Lattanzi, S., Norouzi-Fard, A., and Svensson, O. Consistent k-clustering for general metrics. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 2660–2678. SIAM, 2021.

Guo, X., Kulkarni, J., Li, S., and Xian, J. Consistent k-median: Simpler, better and robust. In *International Conference on Artificial Intelligence and Statistics*, pp. 1135–1143. PMLR, 2021.

Jaghargh, M. R. K., Krause, A., Lattanzi, S., and Vassilvtiskii, S. Consistent online optimization: Convex and submodular. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 2241–2250. PMLR, 2019.

Kalashnikov, D. V., Chen, Z., Mehrotra, S., and Nuray-Turan, R. Web people search via connection analysis. *IEEE Transactions on Knowledge and Data Engineering*, 20(11):1550–1565, 2008.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Lattanzi, S. and Vassilvitskii, S. Consistent k-clustering. In *International Conference on Machine Learning*, pp. 1975–1984. PMLR, 2017.

Lattanzi, S., Moseley, B., Vassilvitskii, S., Wang, Y., and Zhou, R. Robust online correlation clustering. In *NeurIPS*, 2021.

Leskovec, J. and Krevl, A. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.

Mathieu, C., Sankur, O., and Schudy, W. Online correlation clustering. In *27th International Symposium on Theoretical Aspects of Computer Science, STACS 2010, March 4-6, 2010, Nancy, France*, pp. 573–584, 2010.

Algorithm 3 AGREEMENT

```
1: Input: A graph $G$, and a parameter $\epsilon$
2: Output: A clustering $\mathcal{C}$ of $G$
3: Initialize: $\hat{G} \leftarrow G$
4: Step 1: Remove all edges of $\hat{G}$ whose endpoints are not in $\epsilon$-agreement in $G$.
5: Step 2: Remove all edges of $\hat{G}$ whose endpoints are both light.
6: Step 3: Compute the connected components $\mathcal{C}$ of $\hat{G}$.
7: return $\mathcal{C}$
```

A. Preliminaries

In all the subsequent sections we will convert, for simplicity, a complete undirected signed graph $G' = (V, E', s)$ into a non-signed undirected graph $G = (V, E)$ where for each pair of nodes $\{u, v\}$ there is an edge between them in G if and only if $s(\{u, v\}) =' +'$. Thus the absence of an edge between two nodes corresponds to a negative edge in the original signed graph and the presence of an edge to a positive edge in the original graph. Note that the correlation clustering cost of an assignment function f and the clustering $\mathcal{C}$ induced by f becomes:

$$\text{cost}(f) = \sum_{\substack{\{u,v\}\in E \\ f(u)\neq f(v)}} 1 + \sum_{\substack{\{u,v\}\notin E \\ f(u)=f(v)}} 1$$

In addition, the *Agreement* algorithm 1 presented in the main paper needs to be redefined in order to take as input a non-signed graph instead of a signed one. Thus, when we refer to the *Agreement* algorithm we will refer to Algorithm 3.

B. Properties of the *Agreement* algorithm

In this section we will restate some useful definitions and lemmas from (Cohen-Addad et al., 2021) regarding the *Agreement* algorithm and prove some additional structural properties of the clustering produced by the latter. For simplicity, we set both parameters λ, β of the *Agreement* algorithm to be $\lambda = \beta = \epsilon$. We will denote the clustering computed by the *Agreement* algorithm on a graph G as $\tilde{\mathcal{C}}$ and by $\tilde{f}$ the assignment function that induces the latter clustering[5].

A cluster $C \in \tilde{\mathcal{C}}$ will be called non-trivial/non-singleton if $|C| \geqslant 2$ and singleton/trivial otherwise. In addition, we will denote by $f^{\mathcal{O}}$ the assignment function that induces the optimal correlation clustering solution $\mathcal{O}$ of a graph G.

For completeness, we start by restating the two basic definitions used by the *Agreement* algorithm.

Definition 4. *Two nodes u, v are in ϵ-agreement in G if $|N_G(u) \triangle N_G(v)| < \epsilon \max\{|N_G(u)|, |N_G(v)|\}$, where the symbol $\triangle$ denotes the symmetric difference of two sets.*

Definition 5. *A node u is light if it is in ϵ-agreement with less than an ϵ fraction of its neighborhood.*

As a direct consequence of the two latter definitions 1, 2 we have that:

Proposition B.1. *If u, v are in ϵ-agreement then $|N_G(u) \cap N_G(v)| \geqslant (1-\epsilon) max\{|N_G(u)|, |N_G(v)|\}$*

Note that the latter proposition also implies that nodes which are in ϵ-agreement have similar degrees.

Proposition B.2. *If u, v are in ϵ-agreement then $(1-\epsilon)|N_G(u)| \leqslant |N_G(v)| \leqslant \frac{|N_G(u)|}{1-\epsilon}$*

An important property of clustering $\tilde{\mathcal{C}}$ which will permit us to bound the approximation ratio of our solution, is that its cost constitutes a constant factor approximation to the cost of the optimal correlation clustering. That is:

Lemma 6 (rephrased from (Cohen-Addad et al., 2021))**.** *Let $\tilde{\mathcal{C}} = \text{Agreement}(G, \epsilon)$ then for ϵ small enough $\text{cost}(\tilde{f}) \leqslant \Theta(1)\,\text{cost}(f^{\mathcal{O}})$.*

The following lemmas prove that the non-trivial clusters of $\tilde{\mathcal{C}}$ form dense subgraphs in the initial graph G with a small number of outgoing edges.

[5]Note that a clustering may be induced by many different assignment functions, but two different clusterings cannot be induced by the same assignment function.

Lemma B.3 (rephrased from (Cohen-Addad et al., 2021))**.** *Let C be a non-trivial cluster of $\tilde{\mathcal{C}}$ and u a node which belongs to C. Then $|N_G(u) \cap C| \geqslant (1-9\epsilon)|C|$.*

Lemma B.4 (rephrased from (Cohen-Addad et al., 2021))**.** *Let C be a non-trivial cluster of $\tilde{\mathcal{C}}$ and u, v two nodes which belong to C. Then u and v are in an 5ϵ-agreement. If, in addition, there exists a node $w \in C$ such that both u, v are in ϵ-agreement with w then u and v are in an 3ϵ-agreement.*

Note that by combining the latter Lemma B.4 with Propositions B.1 and B.2 we can conclude that nodes in the same cluster $C \in \tilde{\mathcal{C}}$ have similar neighborhoods.

While lemma B.3 proves that the size of the intersection of a node's neighborhood $N_G(u)$ with the cluster C to which it belongs is lower bounded by $(1-\Theta(\epsilon))|C|$ the following lemma also proves that it is lower bounded by $(1-\Theta(\epsilon))|N_G(u)|$.

Lemma 7. *Let C be a non-trivial cluster of $\tilde{\mathcal{C}}$ then for a small enough constant ϵ it holds that:*

1. *If $|C| \geqslant 10$ then every node in C has at least a $(1-3\epsilon)$-fraction of its edges in G to nodes in C.*
2. *If $|C| < 10$ then nodes in C form a clique with no outgoing edges in G.*

Proof. In this proof, we will heavily rely on the structure of $\hat{G}$, the intermediate graph used by the *Agreement* algorithm. To that end, note that any cluster $C \in \tilde{\mathcal{C}}$ corresponds to a connected component of $\hat{G}$. In addition, note that to derive $\hat{G}$ from G the *Agreement* algorithm deletes edges between light nodes, thus any non-trivial cluster of $\tilde{\mathcal{C}}$ (which corresponds to a connected component of $\hat{G}$) contains at least one heavy node and any light node is connected in $\hat{G}$ (and consequently in ϵ-agreement) with at least one heavy node of the same cluster. We prove the first part of the lemma by distinguishing between two cases, i.e. if u is heavy or if u is a light node.
If u is a heavy node of C then by definition u is in ϵ-agreement with at least an $(1-\epsilon)$ fraction of its neighborhood. Consequently, since edges between a heavy and a light node are not deleted, $|N_G(u) \cap C| \geqslant (1-\epsilon)|N_G(u)|$.
If u is light then let $v \in N_{\hat{G}}(u) \cap C$ be a heavy neighboring node with whom u is in ϵ-agreement. Since u and v are in an ϵ-agreement from Proposition B.1 we get that:

$$|N_G(u) \cap N_G(v)| \geqslant (1-\epsilon)\max\{|N_G(u)|, |N_G(v)|\} \Rightarrow \tag{1}$$
$$|(N_G(u) \cap C) \cap (N_G(v) \cap C)| + |(N_G(u) \setminus C) \cap (N_G(v) \setminus C)| \geqslant (1-\epsilon)\max\{|N_G(u)|, |N_G(v)|\} \Rightarrow \tag{2}$$
$$|(N_G(u) \cap C) \cap (N_G(v) \cap C)| \geqslant (1-\epsilon)\max\{|N_G(u)|, |N_G(v)|\} - |(N_G(u) \setminus C) \cap (N_G(v) \setminus C)| \Rightarrow \tag{3}$$
$$|N_G(u) \cap C| \geqslant (1-\epsilon)\max\{|N_G(u)|, |N_G(v)|\} - |N_G(v) \setminus C| \Rightarrow \tag{4}$$
$$|N_G(u) \cap C| \geqslant (1-\epsilon)\max\{|N_G(u)|, |N_G(v)|\} - \epsilon|N_G(v)| \Rightarrow \tag{5}$$
$$|N_G(u) \cap C| \geqslant (1-\epsilon)|N_G(u)| - \frac{\epsilon}{1-\epsilon}|N_G(u)| \Rightarrow \tag{6}$$
$$|N_G(u) \cap C| \geqslant (1-3\epsilon)|N_G(u)| \tag{7}$$

Where:

1. from line (4) to line (5) we used the fact that node v is heavy , hence the set of nodes with which v is not in ϵ-agreement (a) is a superset of $N_G(v) \cap C$; and (b) is at most $\epsilon|N_G(v)|$;
2. from line (5) to line (6) we used that since u, v are in ϵ-agreement, from proposition B.2 we get that $N_G(v) \leqslant \frac{N_G(u)}{1-\epsilon}$; and
3. from line (6) to line (7) we use that $(1-\epsilon-\frac{\epsilon}{1-\epsilon}) > (1-3\epsilon)$ holds for ϵ small enough.

We proceed into proving the second part of the lemma. Note that from the first part of the lemma we get that for every node $u \in C$, $|N_G(u)| \leqslant \frac{|C|}{1-3\epsilon}$. In order to prove that C forms a clique, assume towards a contradiction that there exist two nodes $u, v \in C$ such that there is no edge (u, v) in G. W.l.o.g. we can assume that u, v have a common neighbor in $\hat{G}$ (otherwise no pair of nodes which belong to C would have a common neighbor and C would be empty) and consequently from lemma B.4 they are in an 3ϵ-agreement. Thus, we get:

$$|N_G(u) \triangle N_G(v)| \leqslant 3\epsilon\max\{|N_G(u)|, |N_G(v)|\} \leqslant 3\epsilon\frac{|C|}{1-3\epsilon} < 1 \tag{1}$$

which is a contradiction for all $\epsilon < 1/31$ whenever $|C| \leqslant 9$.
Now, it remains to prove that whenever $|C| \leqslant 9$ there is no outgoing edge in G from a node in C to a node which is not in C. By using $N_G(u) \leqslant \frac{|C|}{1-3\epsilon}$ we can deduce that whenever $|C| \leqslant 9$ and $\epsilon < 1/31$ we have $|N_G(u)| = |C|$. The latter together with the fact that any two nodes $u, v \in C$ have an edge between them is enough to deduce that $N_G(u) = C$ and C has no outgoing edges. □

At that point it is important to recapitulate all the lemmas concerning non-trivial cluster of $\tilde{\mathcal{C}}$ and their consequences. To that end, let u, v be two nodes which belong to the same non-trivial cluster C of $\tilde{\mathcal{C}}$, then for ϵ small enough the following properties hold.

Property 1 $|N_G(u) \cap C| \geqslant (1-3\epsilon)|N_G(u)|$

Property 2 $|N_G(u) \setminus C| < 3\epsilon|N_G(u)|$

Property 3 $|C| \geqslant (1-3\epsilon)|N_G(u)|$

Property 4 $|N_G(u) \cap C| \geqslant (1-9\epsilon)|C|$

Property 5 $|C \setminus N_G(u)| < 9\epsilon|C|$

Property 6 $|N_G(u)| \geqslant (1-9\epsilon)|C|$

Property 7 $|N_G(u) \cap N_G(v)| \geqslant (1-5\epsilon)\max\{|N_G(u)|, |N_G(v)|\}$

Property 8 $|N_G(v)|(1-5\epsilon) \leqslant |N_G(u)| \leqslant \frac{|N_G(v)|}{1-5\epsilon}$

Property 9 $|C \setminus N_G(u)| < 9\epsilon|C| < \frac{9\epsilon}{1-9\epsilon}|N_G(u)|$

Property 10 $|N_G(u) \setminus C| < 3\epsilon|N_G(u)| < \frac{3\epsilon}{1-3\epsilon}|C|$

Property 11 $N_G(u) \cap N_G(v) \neq \emptyset$

Note that Properties 2, 3, 5, 6, 9, 10, 11 are straightforward consequences of Properties 1, 4, 7, 8 which are stated in the lemmas and propositions of this section.

C. Dynamic Analysis of the clustering sequence $\widetilde{\mathcal{C}}_1, \widetilde{\mathcal{C}}_2, \ldots, \widetilde{\mathcal{C}}_t, \ldots$

The *Agreement* algorithm computes at any new arrival of a node u_t a clustering $\widetilde{\mathcal{C}}_t$ of graph G_t. In the subsequent sections we will bound the competitive ratio of our solution and the worst case recourse of any node. To bound the competitive ratio we prove that our solution at time t, induced by the assignment function f_t, is close to the solution $\widetilde{\mathcal{C}}_t$ induced by $\tilde{f}_t$. To bound the recourse we will ensure that for any node u most of the time $f_t(u) = f_{t-1}(u)$. Thus it is clear that for the latter two facts to be approximately true we need $\tilde{f}_t \approx \tilde{f}_{t-1}$ which, at a high level, is equivalent to prove that the structure of $\widetilde{\mathcal{C}}_t$ is similar to the structure of $\widetilde{\mathcal{C}}_{t-1}$. The current section is devoted to prove the latter. At an intuitive level the lemmas presented in this section prove that:

1. A node which belongs to a non-trivial cluster of $\widetilde{\mathcal{C}}_{t-1}$ will either belong to the same non-trivial cluster in $\widetilde{\mathcal{C}}_t$ or it will form a trivial cluster in $\widetilde{\mathcal{C}}_t$. Thus, a node cannot change non-trivial clusters in consecutive rounds.

2. Two non-trivial clusters of $\widetilde{\mathcal{C}}_{t-1}$ cannot merge in $\widetilde{\mathcal{C}}_t$.

3. A cluster in $\widetilde{\mathcal{C}}_{t-1}$ cannot split in two or more non-trivial clusters in $\widetilde{\mathcal{C}}_t$.

Lemma 8. *Let ϵ be a small enough constant and C and C' be non-trivial clusters of $\widetilde{\mathcal{C}}_{t-1}$ and u, v two nodes in C, C' respectively. In $\widetilde{\mathcal{C}}_t$, u and v cannot belong to the same cluster.*

Proof. Assume towards a contradiction that u, v belong to the same cluster of $\widetilde{\mathcal{C}}_t$. Then both C and C' should have been large clusters with more than 9 nodes, as otherwise from Lemma 7 we have that in G_t, u and v have at most 1 common neighbor (the newly arrived node). This, contradicts Property 7 since $|N_{G_t}(u) \cap N_{G_t}(v)| \geqslant (1-5\epsilon)\max\{|N_{G_t}(u)|, |N_{G_t}(v)|\} \geqslant (1-5\epsilon)\cdot 2 > 1$ for ϵ small enough.
Thus, both C and C' have at least 10 nodes each. Given that $C \cap C' = \emptyset$, we can upper bound $|N_{G_{t-1}}(u) \cap N_{G_{t-1}}(v)|$ by $|N_{G_{t-1}}(u) \setminus C| + |N_{G_{t-1}}(v) \setminus C'|$ which (by Property 2) is less than $3\epsilon(|N_{G_{t-1}}(u)| + |N_{G_{t-1}}(v)|) < 6\epsilon \max\{|N_{G_{t-1}}(u)|, |N_{G_{t-1}}(v)|\}$. In addition, we have $|N_{G_t}(u) \cap N_{G_t}(v)| \leqslant |N_{G_{t-1}}(u) \cap N_{G_{t-1}}(v)| + 1$ and by Property 7 $|N_{G_t}(u) \cap N_{G_t}(v)| \geqslant (1-5\epsilon)\max\{|N_{G_t}(u)|, |N_{G_t}(v)|\}$, By combining the upper and lower bounds on $|N_{G_t}(u) \cap N_{G_t}(v)|$ we get that $(1-11\epsilon)\max\{|N_{G_t}(u)|, |N_{G_t}(v)|\} < 1$ which is false for ϵ small enough. □

Lemma 9. *Let ϵ be a small enough constant and let C be a cluster in $\widetilde{\mathcal{C}}_{t-1}$ and v_t the newly arrived node at time t, then for any two distinct non-trivial clusters C_1 and C_2 in $\widetilde{\mathcal{C}}_t$ either $C_1 \cap C = \emptyset$ or $C_2 \cap C = \emptyset$.*

Proof. If C is singleton, the statement is trivially true as $C_1 \cap C_2 = \emptyset$. Next, we assume the contrary. Towards a contradiction assume that $C_1 \cap C \neq \emptyset$ and $C_2 \cap C \neq \emptyset$. Let $u \in C \cap C_1$ and $v \in C \cap C_2$. Then, by Property 11 u and v have a common neighbor in G_{t-1} and, hence, both clusters C_1 and C_2 of $\widetilde{\mathcal{C}}_t$ have an outgoing edge from C_1 and C_2, respectively, in G_t. By Lemma 7, we conclude that $|C_1| \geqslant 10, |C_2| \geqslant 10$. Since $C_1 \cap C_2 = \emptyset$ we can bound $|N_{G_t}(u) \cap N_{G_t}(v)|$ by $|N_{G_t}(u) \setminus C_1| + |N_{G_t}(v) \setminus C_2|$ which (by Property 2) is at most $3\epsilon(|N_{G_t}(u)| + |N_{G_t}(v)|) < 6\epsilon\max\{|N_{G_t}(u)|, |N_{G_t}(v)|\} \leqslant 6\epsilon\max\{|N_{G_{t-1}}(u)|, |N_{G_{t-1}}(v)|\} + 6\epsilon$.

At the same time $|N_{G_{t-1}}(u) \cap N_{G_{t-1}}(v)| \geqslant (1-5\epsilon)\max\{|N_{G_{t-1}}(u)|, |N_{G_{t-1}}(v)|\}$ from Property 7. By noting that $|N_{G_t}(u) \cap N_{G_t}(v)| \geqslant |N_{G_{t-1}}(u) \cap N_{G_{t-1}}(v)|$ we get:

$$6\epsilon \geqslant (1-11\epsilon)\max\{|N_{G_{t-1}}(u)|, |N_{G_{t-1}}(v)|\}$$

which is false for ϵ small enough. □

Lemma 10. *Let ϵ be a small enough constant and let C', C be two clusters in $\widetilde{\mathcal{C}}_{t-1}$, $\widetilde{\mathcal{C}}_t$ respectively. If $C \cap C' \neq \emptyset$ then $|C| < 2|C'|$.*

Proof. Let $u \in C \cap C'$. Note that $|N_{G_t}(u)| \leqslant |N_{G_{t-1}}(u)| + 1$. In addition, from Property 3 we get that $|N_{G_{t-1}}(u)| \leqslant |C'|/(1-3\epsilon)$ and from Property 6 we get that $|N_{G_t}(u)]| \geqslant (1-9\epsilon)|C|$. By combining the latter inequalities we get that $(1-9\epsilon)|C| \leqslant |C'|/(1-3\epsilon) + 1$, from which we can conclude that $|C| < 2|C'|$ for ϵ small enough. □

Proposition C.1. *Let ϵ be a small enough constant, let C', C be two non-trivial clusters in $\widetilde{\mathcal{C}}_{t-1}$ and $\widetilde{\mathcal{C}}_t$, respectively, and v_t the newly arrived node at time t. Then, if $C \cap C' \neq \emptyset$ we have that all nodes in $C \setminus (C' \cup \{v_t\})$ form trivial clusters in $\widetilde{\mathcal{C}}_{t-1}$ and $|C \setminus C'| \leqslant |C|/2$.*

Proof. Note that if $|C|$ has size 2 the lemma follow trivially, so we assume that $|C| \geqslant 3$.

The fact that all nodes in $C \setminus (C' \cup \{v_t\})$ form trivial clusters in $\widetilde{\mathcal{C}}_t$ is a direct consequence of Lemma 8, since nodes from two different non-trivial clusters of $\widetilde{\mathcal{C}}_{t-1}$ cannot be in agreement in the same cluster of $\widetilde{\mathcal{C}}_t$. To prove that $|C \setminus C'| \leqslant |C|/2$ we argue that the intersection of C, C' must be large, i.e., it suffices to argue that $|C \cap C'| > |C|/2$.

Let $u \in C \cap C'$. Then from Property 1 we have that $|N_{G_{t-1}}(u) \cap C'| \geqslant (1-3\epsilon)|N_{G_{t-1}}(u)|$ and $|N_{G_t}(u) \cap C| \geqslant (1-3\epsilon)|N_{G_t}(u)|$. In addition, note that $N_{G_{t-1}}(u) \cap C' = N_{G_t}(u) \cap C'$, hence $|N_{G_t}(u) \cap C'| \geqslant (1-3\epsilon)|N_{G_{t-1}}(u)| \geqslant (1-3\epsilon)(|N_{G_t}(u)| - 1)$. By combing these inequalities we get: $|N_{G_t}(u) \cap C \cap C'| \geqslant (1-6\epsilon)|N_{G_t}(u)| - 1 + 3\epsilon \geqslant (1-9\epsilon)(1-6\epsilon)|C| - 1 + 3\epsilon > |C|/2$ where in the second inequality we used Property 6 and in the last inequality the fact that $|C| \geqslant 3$ and that the inequality $(1-9\epsilon)(1-6\epsilon)x - 1 + 3\epsilon > x/2$ is true for $x \geqslant 3$ and ϵ small enough. □

D. Main Algorithm Notation

Before proceeding, for completeness, we introduce again some auxiliary notation and definitions which are used by the *Online Agreement* Algorithm 2. We denote by $\widetilde{\mathcal{C}}_t$ the clustering computed by the *Agreement* algorithm and by $\mathcal{C}_t$ the clustering solution computed by *Online Agreement* at time t. In addition, we denote by $f_t, \widetilde{f}_t$ the assignment functions which induce the clusterings $\mathcal{C}_t, \widetilde{\mathcal{C}}_t$, respectively. With a slight abuse of notation we denote by $f_t(S), \widetilde{f}_t(S)$ the common cluster id assigned to a set of nodes S by the respective assignment functions.

Let $\mathcal{C}$ be a clustering. We use $S(\mathcal{C}), H(\mathcal{C})$ to denote the set of nodes that belong to trivial and non-trivial, respectively, clusters of $\mathcal{C}$.

As underlined in the main paper, a crucial definition which permits us to keep track of the growth of an evolving cluster is the *origin cluster* definition:

Definition 11 (Origin cluster)**.** *Let ID be a cluster id that is used by the assignment function f_t, and let t_{min} be the minimum t for which ID is used by $f_{t_{min}}$. The origin cluster of ID, denoted by $Origin(ID)$, is the cluster with id ID in the clustering induced by $f_{t_{min}}$.*

An important observation is that, by construction, for any cluster $C' \in \widetilde{\mathcal{C}}_t$ the function f_t[6] assigns the same cluster id to all nodes in C'. In addition, note that f_t may assign the latter cluster id, i.e., $f_t(C')$, also to nodes which form trivial clusters in $\widetilde{\mathcal{C}}_t$, i.e., nodes in $S(\widetilde{\mathcal{C}}_t)$. For that reason, for a cluster $C' \in \widetilde{\mathcal{C}}_t$ we denote the set of nodes in $S(\widetilde{\mathcal{C}}_t)$ to which f_t assigns cluster id $f_t(C')$ as $S_t^{C'}$, that is:

$$S_t^{C'} = \{u \in S(\widetilde{\mathcal{C}}_t) : f_t(u) = f_t(C')\}$$

We mention the following trivial proposition, which is implied by the *Assignment refinement phase* of *Online Agreement*, as it is heavily used in all of our lemmas.

Proposition D.1. *Let C' be a cluster of $\widetilde{\mathcal{C}}_t$. Then $|C'| < (3/2)|Origin(f_t(C'))|$.*

E. Bound the competitive ratio

Lemma 12. *Let ϵ be a small enough constant and C' a non-trivial cluster of $\widetilde{\mathcal{C}}_t$ such that $Origin(f_t(C')) \cap C' \neq \emptyset$. Then $|Origin(f_t(C')) \cap C'| > (1-20\epsilon)|Origin(f_t(C'))|$.*

Proof. For simplicity, let $C = Origin(f_t(C'))$ and assume w.l.o.g. that C was formed at some time $t_0 < t$ (otherwise the lemma follows trivially since $C = C'$). Let $x \in C \cap C'$. Since $x \in C$, Property 4 gives that $|N_{G_{t_0}}(x) \cap C| \geqslant (1-9\epsilon)|C|$. Thus, at time t the edges of x outside C', i.e., $|N_{G_t}(x) \setminus C'|$ are at least $(1-9\epsilon)|C| - |C \cap C'|$. From Property 2 we get $|N_{G_t}(x) \setminus C'| < 3\epsilon|N_{G_t}(x)|$. By combining the lower and upper bound we get the following inequality:

$$\begin{aligned}(1-9\epsilon)|C| - |C \cap C'| < 3\epsilon|N_{G_t}(x)| \Rightarrow \\ \frac{(1-9\epsilon)|C| - |C \cap C'|}{3\epsilon} < |N_{G_t}(x)|\end{aligned}$$

From Proposition D.1 we know that $|C'| < 3/2|C|$ and from Property 3 we get that $(1-3\epsilon)|N_{G_t}(x)| \leqslant |C'|$. Thus:

$$\begin{aligned}&(1-3\epsilon)|N_{G_t}(x)| < 3/2|C| \Rightarrow \\ &|N_{G_t}(x)| < \frac{3|C|}{2(1-3\epsilon)}\end{aligned}$$

Combining the lower and upper bound on $N_{G_t}(x)$ we get:

$$|C \cap C'| > |C|\left(1 - 9\epsilon - \frac{9\epsilon}{2(1-3\epsilon)}\right) > (1-20\epsilon)|C|$$

for ϵ small enough. □

Lemma 13. *Le ϵ be a small enough constant and C' a non-trivial cluster of $\widetilde{\mathcal{C}}_t$. Then $Origin(f_t(C')) \cap C' \neq \emptyset$.*

Proof. For simplicity, let $C = Origin(f_t(C'))$ and assume w.l.o.g. that C was formed at some time $t_0 < t$ (otherwise the lemma follows trivially since $C = C'$). Assume towards contradiction that C' is the first such cluster which does not intersect with $C = Origin(f_t(C'))$. Since $f_t(C') = f_{t_0}(C)$ there exists a node $u \in C'$ and a cluster C'' of $\widetilde{\mathcal{C}}_{t'}$ for $t' \in [t_0, t)$ such that $C'' \cap C \neq \emptyset$, $u \in C''$ and $f_{t'}(C'') = f_{t_0}(C)$. Indeed if such a node u and cluster C'' do not exist then $f_t(C') \neq f_{t_0}(C)$ because we assumed that C' is the first cluster s.t. $C' \cap C = \emptyset$ and $f_t(C') = f_{t_0}(C)$, and moreover, in

[6]note that also the function $\tilde{f}_t$ assigns the same cluster id to all nodes in C'.

order for C' to be assigned the same cluster id as C it should intersect a non-singleton cluster of $\widetilde{\mathcal{C}}_{t-1}$ which has the same cluster id as C.

We lower bound the number of edges u has towards nodes of C. To that end, note that at time t' from Property 4 it holds that $|N_{G_{t'}}(u) \cap C''| \geqslant (1-9\epsilon)|C''|$. In addition to that, due to the Lemma 12 we have that $|C \cap C''| \geqslant (1-20\epsilon)|C| > (1-20\epsilon)\frac{2}{3}|C''| > \frac{|C''|}{2}$, where the second inequality comes from the fact that $|C''| < 3/2|C|$ and the third inequality is true for ϵ small enough. By combining the last two inequalities we get:

$$\begin{aligned}
|N_{G_{t'}}(u) \cap (C'' \cap C)| &\geqslant |C'' \cap C| - |(C'' \cap C) \setminus N_{G_{t'}}(u)| && (1)\\
&\geqslant |C'' \cap C| - |C'' \setminus N_{G_{t'}}(u)| && (2)\\
&\geqslant |C''|/2 - 9\epsilon|C''| && (3)\\
&> |C''|/3 && (4)\\
&\geqslant \frac{(1-20\epsilon)}{3}|C| && (5)\\
&\geqslant \frac{|C|}{6} && (6)
\end{aligned}$$

Where:

1. from line (3) to line (4) $(1/2 - 9\epsilon) > 1/3$ is true for ϵ small enough;
2. from line (4) to line (5) we used Lemma 12; and
3. from line (5) to line (6) $\frac{(1-20\epsilon)}{3} \geqslant \frac{1}{6}$ is true for ϵ small enough.

Consequently at time t, since $C' \cap C = \emptyset$, $|N_{G_t}(u) \setminus C'| \geqslant |N_{G_t}(u) \cap C| \geqslant |N_{G_t}(u) \cap (C'' \cap C)| > \frac{|C|}{6}$. By combining the latter with $|N_{G_t}(u) \setminus C'| < \frac{3\epsilon}{1-3\epsilon}|C'|$ from Property 10 we get

$$|C'| > \frac{1-3\epsilon}{18\epsilon}|C| > \frac{3}{2}|C|$$

for ϵ small enough.

However, because of Proposition D.1, we get that $|C'| < \frac{3}{2}|C|$, which is a contradiction. This concludes the proof. □

Note that by combining the two latter Lemmas 12 and 13 we can deduce that an origin cluster which gets assigned by an assignment function $f_{t'}$ cluster id s has a large intersection with a cluster of $\widetilde{\mathcal{C}}_t$ whose nodes get assigned cluster id s by an assignment function f_t for $t > t'$. Formally:

Corollary E.1. *Le ϵ be a small enough constant and C' a non-trivial cluster of $\widetilde{\mathcal{C}}_t$. Then $|Origin(f_t(C')) \cap C'| > (1-20\epsilon)|Origin(f_t(C'))|$.*

Using corollary E.1 we can deduce that there cannot be two clusters C_1, C_2 of $\widetilde{\mathcal{C}}_t$ whose nodes get assigned the same cluster id by f_t.

Lemma 14. *Le ϵ be a small enough constant and C', C'' be two non-trivial clusters of $\widetilde{\mathcal{C}}_t$. Then $f_t(C') \neq f_t(C'')$.*

Proof. Towards a contradiction assume that nodes in C', C'' get assigned the same cluster id s by f_t and let $C = Origin(s)$. Then from Corollary E.1 we have that $|C \cap C'| > (1-20\epsilon)|C|$, $|C \cap C''| > (1-20\epsilon)|C|$. In addition, C', C'' are different clusters of the same clustering $\widetilde{\mathcal{C}}_t$, hence they should not intersect, that is $C' \cap C'' = \emptyset$. Thus, we have $|C| \geqslant |C \cap C'| + |C \cap C''| > (2-40\epsilon)|C|$ which is false for ϵ small enough. □

An important quantity we need to bound is the number of nodes which belong to trivial clusters of $\widetilde{\mathcal{C}}_t$ and get assigned by f_t the same cluster id as nodes which belong to non-trivial clusters of $\widetilde{\mathcal{C}}_t$. To that end, Corollary E.1 is crucial. The following lemmas bound the latter quantity.

Lemma 15. *Let ϵ be a small enough constant and C' be a non-trivial cluster of $\widetilde{\mathcal{C}}_t$, $S_t^{C'} = \{u \in S(\widetilde{\mathcal{C}}_t) : f_t(u) = f_t(C')\}$ be the set of nodes that get assigned cluster id $f_t(C')$ by f_t and form trivial clusters in $\widetilde{\mathcal{C}}_t$ and C be the origin cluster which corresponds to cluster id $f_t(C')$, that is $C = Origin(f_t(C'))$. Then every node $u \in S_t^{C'}$ has at least $2|C|/9$ edges to nodes of C and the total number of edges between nodes in $S_t^{C'}$ and $C \cap C'$ is at least $\frac{|S_t^{C'}||C|}{10} > \frac{|S_t^{C'}||C'|}{15}$.*

Proof. Let $t_0 \leqslant t$ be the time when the origin cluster C was formed. By Corollary E.1 we know that $|C \cap C'| > (1-20\epsilon)|C|$ and since by Proposition D.1 it holds that $|C'| < 3/2|C|$ we get that $|C \cap C'| > |C'|/2$ for ϵ small enough.

Let $x \in S_t^{C'}$. Since $f_t(x) = f_{t_0}(C)$ there exists $t' \in [t_0, t]$ and a cluster C'' of $\widetilde{\mathcal{C}}_{t'}$ such that: (1) $f_{t'}(C'') = f_{t_0}(C)$; and (2) $x \in C''$. Again by the Corollary E.1 we get that $|C \cap C''| \geqslant |C''|/2$ for ϵ small enough. Since $x \in C''$ we also have $|N_{G_{t'}}(x) \cap C''| \geqslant (1-9\epsilon)|C''|$. By combining the two latter inequalities we get that:

$$\begin{aligned}
|N_{G_{t'}}(x) \cap (C'' \cap C)| &\geqslant |C'' \cap C| - |(C'' \cap C) \setminus N_{G_{t'}}(x)| && (1)\\
&\geqslant |C'' \cap C| - |C'' \setminus N_{G_{t'}}(x)| && (2)\\
&\geqslant |C''|/2 - 9\epsilon|C''| && (3)\\
&> |C''|/3 && (4)\\
&> 2|C|/9 && (5)
\end{aligned}$$

Where:

1. from line (3) to line (4) we used the fact that $(1/2 - 9\epsilon) > 1/3$ holds for ϵ small enough; and
2. from line (4) to line (5) we used the fact that $|C''| < 3/2|C|$ holds by Proposition D.1.

Consequently, every node of $S_t^{C'}$ has at least $2|C|/9$ edges to nodes of C. Using the latter we can deduce that the total number of edges between nodes of $S_t^{C'}$ and $C \cap C'$ is at least:

$$|S_t^{C'}|\frac{2}{9}|C| - |C \setminus C'||S_t^{C'}| > |S_t^{C'}|\frac{2}{9}|C| - 20\epsilon|C||S_t^{C'}| = |S_t^{C'}||C|\left(\frac{2}{9} - 20\epsilon\right) \geqslant \frac{|S_t^{C'}||C|}{10}$$

Where:

1. the first inequality comes from Corollary E.1; and
2. the last inequality holds since $(\frac{2}{9} - 20\epsilon) \geqslant \frac{1}{10}$ for ϵ small enough.

To conclude the lemma just note that by Proposition D.1 $|C'| < 3/2|C|$ and consequently $\frac{|S_t^{C'}||C|}{10} > \frac{|S_t^{C'}||C'|}{15}$.

□

Lemma 16. *Let ϵ be a small enough constant, C' be a cluster of $\widetilde{\mathcal{C}}_t$ and $S_t^{C'} = \{u \in S(\widetilde{\mathcal{C}}_t) : f_t(u) = f_t(C')\}$ be the set of nodes that get assigned cluster id $f_t(C')$ by f_t and form trivial clusters in $\widetilde{\mathcal{C}}_t$. Then $|S_t^{C'}| < 100\epsilon|C'|$.*

Proof. Let C be the origin cluster corresponding to cluster id $f_t(C')$, that is $C = Origin(f_t(C'))$. By Lemma 15 we know that the total number of edges between nodes of $S_t^{C'}$ and $C \cap C'$ is at least $\frac{|S_t^{C'}||C|}{10}$. Thus, by averaging there is a node $r \in C \cap C'$ with at least $\frac{|S_t^{C'}||C|}{10|C \cap C'|} > \frac{|S_t^{C'}||C|}{10|C|} > \frac{|S_t^{C'}|}{10}$ edges to nodes of $S_t^{C'}$. Thus, for that node r it holds that:

1. $|N_{G_t}(r) \setminus C'| > \frac{|S_t^{C'}|}{10}$; and
2. $|N_{G_t}(r) \setminus C'| < \frac{3\epsilon}{1-3\epsilon}|C'|$ from Property 10

By combining these two bounds we conclude that $|S_t^{C'}| < \frac{30\epsilon}{1-3\epsilon}|C'| < 100\epsilon|C'|$ for ϵ small enough. □

Before stating the main theorem of this section, we underline that by $\mathcal{O}_t$ we denote the optimal correlation clustering solution for graph G_t and by $f^{\mathcal{O}_t}$ the assignment function which induces that clustering.

Theorem E.2. *The Online Agreement Algorithm is a constant competitive ratio algorithm, that is, for any time t*

$$\operatorname{cost}(f_t) \leqslant \Theta(1) \cdot \operatorname{cost}(f^{\mathcal{O}_t})$$

Proof. Fix a time t, from Lemma 6 we have that the cost of clustering $\widetilde{\mathcal{C}}_t$ is a constant factor approximation to the cost of the optimal correlation clustering solution $\mathcal{O}_t$. Thus to prove the theorem it suffices to prove that the cost of our solution is a constant factor approximation to the cost of $\widetilde{\mathcal{C}}_t$. W.l.o.g. we assume that $\operatorname{cost}(\tilde{f}_t) < \operatorname{cost}(f_t)$, otherwise the theorem becomes trivial.

First, note that for any non-trivial cluster C of $\widetilde{\mathcal{C}}_t$ both f_t and $\tilde{f}_t$ cluster all nodes of C together. Now, fix a node u which forms a trivial cluster in $\widetilde{\mathcal{C}}_t$. While $\tilde{f}_t$ clusters u as a singleton cluster f_t may cluster u in a larger cluster. We concentrate on the case where f_t clusters u in a larger cluster C^*, as this is the case where the cost of the two assignment functions may differ. Both assignment functions maintain, at all time, the following invariant: if two nodes v, v' belong to different non-trivial clusters of $\widetilde{\mathcal{C}}_t$ then they are assigned to different clusters in $\mathcal{C}_t$. Thus, C^* may contain the nodes of at most one non-trivial cluster of $\widetilde{\mathcal{C}}_t$.

We start by arguing that if $C^* \subseteq S(\widetilde{\mathcal{C}}_t)$ we can safely charge the cost that f_t pays for clustering all nodes of C^* together to what $\tilde{f}_t$ pays for clustering them apart. To argue the latter, it suffices to prove that $|(u,v) \notin E : u \in C^*, v \in C^*| \leqslant \Theta(1)|(u,v) \in E : u \in C^*, v \in V|$. To that end, note that $C^* \subseteq C' \cup S_{t'}^{C'}$ for some $t' < t$ and non-trivial cluster $C' \in \widetilde{\mathcal{C}}_{t'}$. W.l.o.g. assume t' is the last time the latter holds and let C be the origin cluster corresponding to $f_{t'}(C')$. Then, by Lemma 15 any node in C^* has at least $2|C|/9$ edges to nodes in C, hence $|(u,v) \in E : u \in C^*, v \in C^*| \geqslant |C^*| \cdot 2|C|/9 = \Theta(1)|C||C^*|$. At the same time we have that $|(u,v) \notin E : u \in C^*, v \in C^*| \leqslant |C^*|^2 \leqslant |C^*|(|C'| + |S_{t'}^{C'}|) \leqslant |C^*|(|C'| + 100\epsilon|C'|) = \Theta(1)|C^*||C|$, where we used Lemma 16 and the fact that C is the origin cluster of C'. Thus, for the rest of the proof we can assume that $C^* \nsubseteq S(\widetilde{\mathcal{C}}_t)$ by loosing only a constant factor in the approximation guarantees.

Since $C^* = C \cup S_t^C$ for some non-trivial cluster $C \in \widetilde{\mathcal{C}}_t$, we have that under the assignment function f_t all nodes in $C \cup S_t^C$ are clustered together while under the assignment function $\tilde{f}_t$ nodes in C are clustered together and each node in S_t^C is clustered as a singleton. To relate the two costs $\operatorname{cost}(\tilde{f}_t), \operatorname{cost}(f_t)$ we define the following sets of pairs of nodes and assume that pair of nodes are not ordered so that they are not double counted:

- $PCS(C) = \{(u,v) \in E : u \in C, v \in S_t^C\}$.
- $NCS(C) = \{(u,v) \notin E : u \in C, v \in S_t^C\}$
- $PSS(C) = \{(u,v) \in E : u \in S_t^C, v \in S_t^C\}$
- $NSS(C) = \{(u,v) \notin E : u \in S_t^C, v \in S_t^C\}$
- $PVS(C) = \{(u,v) \in E : u \in V_t \setminus (C \cup S_t^C), v \in S_t^C\}$
- $NVS(C) = \{(u,v) \notin E : u \in V_t \setminus (C \cup S_t^C), v \in S_t^C\}$
- $PCC(C) = \{(u,v) \in E : u \in C, v \in C\}$
- $NCC(C) = \{(u,v) \notin E : u \in C, v \in C\}$

Note that: (1) pairs of nodes in sets $PCS(C), PSS(C)$ contribute to $\operatorname{cost}(\tilde{f}_t)$ but not to $\operatorname{cost}(f_t)$; (2) pairs of nodes in sets $NCS(C), NSS(C)$ contribute to $\operatorname{cost}(f_t)$ but not to $\operatorname{cost}(\tilde{f}_t)$; (3) pairs of nodes in $PVS(C)$ and $NCC(C)$ contribute to both costs; and (4) pairs of nodes in $NVS(C)$ and $PCC(C)$ do not contribute to none of the two costs. In addition, we have that for every two different clusters $C, C' \in \widetilde{\mathcal{C}}_t$ by Lemma 14 the sets $S_t^C, S_t^{C'}$ do not intersect, hence the difference of the two costs can be rewritten as:

$$\begin{aligned}\operatorname{cost}(f_t) - \operatorname{cost}(\tilde{f}_t) &= \sum_{C \in \widetilde{\mathcal{C}}_t : |C| > 1} (|NCS(C)| + |NSS(C)| - |PCS(C)| - |PSS(C)|) \\ &\leqslant \sum_{C \in \widetilde{\mathcal{C}}_t : |C| > 1} (|NCS(C)| + |NSS(C)|)\end{aligned}$$

We are left to prove that $|NCS(C)| \leqslant \Theta(1)|PCS(C)|$ and $|NSS(C)| \leqslant \Theta(1)|PCS(C)|$ for every $C \in \widetilde{\mathcal{C}}_t$. This way we deduce that $\operatorname{cost}(f_t) - \operatorname{cost}(\tilde{f}_t) \leqslant \Theta(1) \sum_{C \in \widetilde{\mathcal{C}}_t : |C|>1} |PCS(C)|$. Note that the latter suffices to prove the Lemma since $\operatorname{cost}(\tilde{f}_t) \geqslant \sum_{C \in \widetilde{\mathcal{C}}_t : |C|>1} |PCS(C)|$.

We lower bound the size of $PCS(C)$ by $\frac{|S_t^C||C|}{15}$ using[7] Lemma 15. In addition from the definition of set $NCS(C)$ we can upper bound its size by $|S_t^C||C|$, hence, deduce that $|NCS(C)| \leqslant 15|PCS(C)|$. Finally, note that for set $NSS(C)$ we have:

$$|NSS(C)| \leqslant |S_t^C|^2 < 100\epsilon |S_t^C||C| \leqslant 100\epsilon |PSC(C)|$$

where the first inequality comes from the definition of set $NSS(C)$, the second inequality from Lemma 16 and the third inequality from the lower bound on the size of $PSC(C)$. This concludes the proof. □

F. Bound the worst case recourse

In this section we bound the worst case recourse of any node by $O(\log(n))$ where n is the number of nodes in the final graph. To do the latter we prove that whenever a node changes from a non-trivial cluster $C_1' \in \widetilde{\mathcal{C}}_{t_1}$ whose nodes get assigned cluster id s_1 by f_{t_1} to another non-trivial cluster $C_2' \in \widetilde{\mathcal{C}}_{t_2}$ whose nodes get assigned cluster id s_2 by f_{t_2} then the origin cluster corresponding to the new cluster id is bigger than the one corresponding to the old cluster id by a multiplicative term. Since the maximum size of an origin cluster is n, changing non–trivial clusters and consequently changing the origin cluster corresponding to the assigned cluster id can happen at most $\log(n)$ times.

We will first prove that if two origin clusters are "close" in size then they cannot be intersecting.

Lemma 17. *Let ϵ be a small enough constant and let C_1, C_2 be two origin clusters with cluster ids s_1, s_2 formed at different times t_1, t_2 respectively with $t_1 < t_2$. If $|C_2| < 5/4|C_1|$ then $C_1 \cap C_2 = \emptyset$.*

Proof. Towards a contradiction assume that $C_1 \cap C_2 \neq \emptyset$ and let $u \in C_1 \cap C_2$. As a first step of the proof we will argue that since $C_1 \cap C_2$ is non-empty then $|C_1 \cap C_2|$ must be large. Since $u \in C_1 \cap C_2$ from Property 4 we have that $|N_{Gt_1}(u) \cap C_1| > (1 - 9\epsilon)|C_1|$. Thus, $|N_{G_{t_2}}(u) \setminus C_2| \geqslant (1 - 9\epsilon)|C_1| - |C_1 \cap C_2|$. Combining the latter with the upper bound on $|N_{G_{t_2}}(u) \setminus C_2|$ from Property 10 we get:

$$(1 - 9\epsilon)|C_1| - |C_1 \cap C_2| < \frac{3\epsilon}{1 - 3\epsilon}|C_2| \Rightarrow \tag{1}$$

$$(1 - 9\epsilon)|C_1| - \frac{3\epsilon}{1 - 3\epsilon}|C_2| < |C_1 \cap C_2| \Rightarrow \tag{2}$$

$$(1 - 9\epsilon)|C_1| - \frac{15\epsilon}{4(1 - 3\epsilon)}|C_1| < |C_1 \cap C_2| \Rightarrow \tag{3}$$

$$(1 - 20\epsilon)|C_1| < |C_1 \cap C_2| \tag{4}$$

Where:

1. from line (2) to line (3) we used that $|C_2| < 5/4|C_1|$; and

2. from line (3) to line (4) we used that $(1 - 9\epsilon) - \frac{15\epsilon}{4(1-3\epsilon)} > (1 - 20\epsilon)$ is true for ϵ small enough.

[7]Note that cluster C corresponds to cluster C' in Lemma 15

Now, we will argue that at time $t_2 - 1$ there should be a node $v \in C_1 \cap C_2$ such that $f_{t_2-1}(v) \neq s_1$.

Let v_{t_2} be the newly arrived node at time t_2 and note that because of lemma 8, C_2 is formed either:

Case 1: exclusively from nodes in $S(\widetilde{\mathcal{C}}_{t_2-1}) \cup \{v_{t_2}\}$ (by using **Rule 2**); or

Case 2: by a subset of nodes $C' \neq \emptyset$ in a non-trivial cluster $C'' \in \widetilde{\mathcal{C}}_{t_2-1}$ and nodes in $S(\widetilde{\mathcal{C}}_{t_2-1}) \cup \{v_{t_2}\}$.

We will continue by proving that in neither of these two cases all nodes in $C_1 \cap C_2$ can have the same cluster id s_1 at time $t_2 - 1$.

Before doing so, note that by combining $(1-20\epsilon)|C_1| < |C_1 \cap C_2|$ and $|C_2| < 5/4|C_1|$ we can deduce that $(4/5)(1-20\epsilon)|C_2| < |C_1 \cap C_2|$ and consequently $|C_1 \cap C_2| > |C_2|/2$ for ϵ small enough.

In **Case 1** where C_2 is formed exclusively by nodes in $S(\widetilde{\mathcal{C}}_{t_2-1}) \cup \{v_{t_2}\}$, if all nodes $v \in C_1 \cap C_2$ have a cluster id s_1 at time $t_2 - 1$ then, since $|C_1 \cap C_2| > |C_2|/2$ by **Rule 2** of Algorithm 2 the new cluster would have had s_1, and not s_2, as a cluster id before the ***Assignment refinement phase***. In addition, since $|C_2| < 5/4|C_1| < 3/2|C_1|$ the assignment function f_{t_2} will still assign s_1 after the ***Assignment refinement phase***, which is a contradiction. Thus, in this case all nodes in $C_1 \cap C_2$ cannot have the same cluster id s_1 at time $t_2 - 1$.

In **Case 2** note that all nodes in C' at time $t_2 - 1$ have a cluster id different from s_1, otherwise f_{t_2} would have assigned cluster id s_1 to all nodes in C_2 before the ***Assignment refinement phase*** and also after the ***Assignment refinement phase*** since $|C_2| < 5/4|C_1| < 3/2|C_1|$. Thus, since in **Case 2** C' is non-empty,

- either $C_1 \cap C_2 \cap C'$ is non-empty, and consequently there is a node in $C_1 \cap C_2 \supseteq C_1 \cap C_2 \cap C'$, whose cluster id at time $t_2 - 1$ is different from s_1; or
- $C_1 \cap C_2 \cap C'$ is empty and all nodes in $C_1 \cap C_2$ form trivial clusters in $\widetilde{\mathcal{C}}_{t_2-1}$. In that case, as in **Case 1** there should be a node in $C_1 \cap C_2$ whose id at time $t_2 - 1$ is different than s_1, otherwise the Algorithm 2 would have assigned the same cluster id s_1 to C_2.

Consequently, there exists a node $v \in C_1 \cap C_2$ such that $f_{t_2-1}(v) \neq s_1$.

Since there exists a node $v \in C_1 \cap C_2$ such that $f_{t_2-1}(v) \neq s_1$, w.l.o.g. we can assume that v was the last node to change cluster id before time t_2. Let $C''' \in \widetilde{\mathcal{C}}_{t'}$ be last non-trivial cluster to which v belonged for $t' \in (t_1, t_2 - 1)$ before changing cluster id. We proceed by bounding the size of C'''. From Property 3 $|N_{G_{t_2}}(v)| \leqslant \frac{|C_2|}{1-3\epsilon} < \frac{5}{4}\frac{1}{1-3\epsilon}|C_1|$, hence $|N_{G_{t_2}}(v)| < \frac{5}{4}\frac{1}{1-3\epsilon}|C_1|$. At the same time from Property 6 $|N_{G_{t_2}}(v)| \geqslant |N_{G_{t'}}(v)| > (1-9\epsilon)|C'''|$, thus

$$|C'''| < \frac{1}{1-9\epsilon}\frac{5}{4}\frac{1}{1-3\epsilon}|C_1| < \frac{4}{3}|C_1|$$

for ϵ small enough. Now using the same arguments as in the beginning of the proof where we lower bounded the size of $C_1 \cap C_2$ we can argue that $|C''' \cap C_1| > (1-20\epsilon)|C_1|$ for ϵ small enough. Consequently, because w.l.o.g. we assumed that v is the last node of $C_1 \cap C_2$ to change cluster id, we have that $f_{t_2-1}(w) = f_{t'}(C'''), \forall w \in C_1 \cap C_2 \cap C'''$. In addition note that from $|C_1 \cap C_2| > (1-20\epsilon)|C_1|$ and $|C''' \cap C_1| > (1-20\epsilon)|C_1|$ we can deduce:

$$|C_1 \cap C_2 \cap C'''| > (1-40\epsilon)|C_1| > \frac{4}{5}(1-40\epsilon)|C_2| > \frac{|C_2|}{2}$$

where the last inequality holds for ϵ small enough.

Let C'''' be the origin cluster corresponding to $f_{t'}(C''')$. From corollary E.1 we know that $|C''''| < \frac{|C'''|}{(1-20\epsilon)}$. Combining that with $|C_1 \cap C_2 \cap C'''| > \frac{4}{5}(1-40\epsilon)|C_2| \Rightarrow \frac{5}{4(1-40\epsilon)}|C'''| > |C_2|$. We conclude that $|C_2| < (3/2)|C''''|$.

Again, we will continue the proof by distinguishing between two cases:

Case 1: where C_2 is formed exclusively by nodes in $S(\widetilde{\mathcal{C}}_{t_2-1}) \cup \{v_{t_2}\}$; and

Case 2: where C_2 is formed by a subset of nodes $C' \neq \emptyset$ in a non-trivial cluster $C'' \in \widetilde{\mathcal{C}}_{t_2-1}$ and nodes in $S(\widetilde{\mathcal{C}}_{t_2-1}) \cup \{v_{t_2}\}$.

In **Case 1** we have that C_2 is formed by nodes in $S(\widetilde{\mathcal{C}}_{t_2-1}) \cup \{v_t\}$ where at least half of the nodes have cluster id $f_{t'}(C''')$ at time $t_2 - 1$. By **Rule 2** all nodes in C_2 get as a cluster id $f_{t'}(C''')$ before the ***Assignment refinement phase*** and remain with that cluster id also after the ***Assignment refinement phase*** since $|C_2| < (3/2)|C''''|$. Thus all nodes in C_2 get cluster id $f_{t'}(C''')$ and C_2 is not an origin cluster in that case, which is a contradiction.

In **Case 2**, we can argue again that nodes in C' cannot have as a cluster id $f_{t'}(C''')$ since $|C_2| < (3/2)|C''''|$ and C_2 would not be an origin cluster. Thus, all nodes of C_2 whose cluster id is $f_{t'}(C''')$ at time $t_2 - 1$ form trivial clusters in $\widetilde{\mathcal{C}}_{t_2-1}$. From proposition C.1 such nodes are at most $|C_2|/2$ which contradicts the fact that $|C_1 \cap C_2 \cap C'''| > \frac{|C_2|}{2}$.

Which leads to a contradiction. The lemma follows. □

Lemma 18. *Let node v be a node which belongs to a non-trivial cluster $C_1' \in \widetilde{\mathcal{C}}_{t_1}$ and to a non-trivial cluster $C_2' \in \widetilde{\mathcal{C}}_{t_2}$ for $t_1 < t_2$. If $f_{t_1}(C_1') \neq f_{t_2}(C_2')$ then denote by C_1 and C_2 the origin clusters corresponding to cluster ids $f_{t_1}(C_1')$ and $f_{t_2}(C_2')$ respectively. Then $|C_2| > (5/4)|C_1|$.*

Proof. Towards a contradiction assume that $|C_2| < 5/4|C_1|$.

From lemma 12 we have that $|C_1 \cap C_1'| \geqslant (1 - 20\epsilon)|C_1|$ and from Proposition D.1 we also have that $|C_1'| < (3/2)|C_1|$. Combining the latter two inequalities we get $|C_1 \cap C_1'| \geqslant \frac{|C_1'|}{2}$ for ϵ small enough. Note that $|N_{G_{t_1}}(v) \cap (C_1' \cap C_1)| \geqslant |N_{G_{t_1}}(v) \cap C_1'| - |C_1' \setminus C_1| \geqslant (1 - 9\epsilon)|C_1'| - |C_1'|/2 > |C_1'|/3 > |C_1|/4$ where: (1) in the second inequality we used Property 6 and the fact that $|C_1 \cap C_1'| > |C_1'|/2$; (2) in the third inequality we used that $(1 - 9\epsilon - 1/2) > 1/3$ for ϵ small enough; and (3) in the last inequality we used that $|C_1| < \frac{|C_1'|}{1-20\epsilon}$ from lemma 12 and $(1 - 20\epsilon)/3 > 1/4$ for ϵ small enough.

We will continue by upper bounding $|N_{G_{t_2}}(v) \setminus C_2|$. To that end from lemma 12 and following the same reasoning as we did to bound $|C_1 \cap C_1'|$ we get that $|C_2 \cap C_2'| \geqslant |C_2'|/2$. In addition:

$$|N_{G_{t_2}}(v) \setminus C_2| \leqslant |N_{G_{t_2}}(v) \setminus C_2 \cap C_2'| \leqslant |N_{G_{t_2}}(v) \setminus C_2'| + |C_2' \setminus C_2| \leqslant \frac{3\epsilon}{1 - 3\epsilon}|C_2'| + |C_2'|/2 < \\ < \left(\frac{3\epsilon}{1 - 3\epsilon} + \frac{1}{2}\right)\frac{|C_2|}{1 - 20\epsilon}$$

Where in the third inequality we used Property 10 and in the last lemma 12.

From lemma 17 we have that $C_1 \cap C_2 = \emptyset$ and $N_{G_{t_1}}(v) \subseteq N_{G_{t_2}}(v)$ so $|N_{G_{t_1}}(v) \cap C_1| \leqslant |N_{G_{t_2}}(v) \setminus C_2|$. Combining the lower and upper bound of the latter two quantities, for ϵ small enough we end up in a contradiction.

$$|C_1|/4 < \left(\frac{3\epsilon}{1 - 3\epsilon} + \frac{1}{2}\right)\frac{|C_2|}{1 - 20\epsilon} \Rightarrow |C_2| > 5/4|C_1|$$

Which leads to a contradiction. □

Theorem F.1. *The Online Agreement Algorithm has a worst case recourse of $O(\log n)$, that is:*

$$r(u) = O(\log n), \forall v \in V$$

Proof. Fix a node u and assume that at time t node u belongs to a non-trivial cluster $C \in \widetilde{\mathcal{C}}_t$ with cluster id s assigned by f_t. Note that node u will get assigned a new cluster id by an assignment function $f_{t''}$ whenever one of the following scenarios happen:

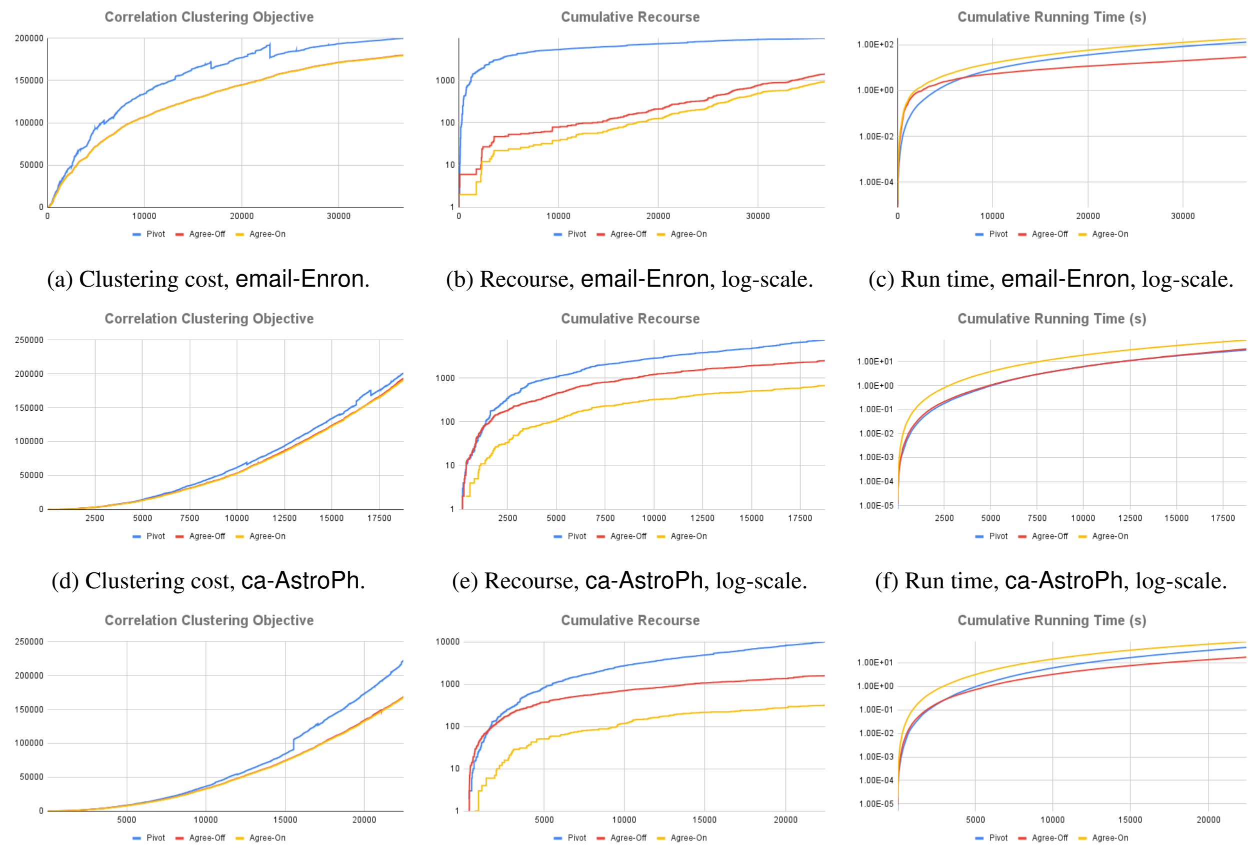
1. node u forms a trivial cluster in $\widetilde{\mathcal{C}}_{t'}$ for $t' > t$ and at a later time $t'' > t'$ enters a non-trivial cluster in $\widetilde{\mathcal{C}}_{t''}$ with a different cluster id s'; and

2. node u forms a trivial cluster in $\widetilde{\mathcal{C}}_{t'}$ for $t' > t$ and at a later time $t'' > t'$ its cluster id changes to s' by the ***Assignment refinement phase***.

Note that (1) can happen at most $O(\log n)$ times due to lemma 18 and each time it happens the recourse of u increases by 1. In addition, note that after (2) happens there are two possibilities; either u remains in a trivial cluster and it never changes its cluster id again or it enters a new non-trivial cluster with cluster id s''. The former can happen at most once and increases the recourse by 1, while the latter increases the recourse by 2 and by lemma 18 can happen at most $O(\log n)$ times. Thus the overall recourse of u is $O(\log n)$. $\square$

G. Omitted experiments

Due to the artificial random arrival sequence that we used for the datasets musae-facebook, ca-AstroPh, email-Enron, the first part of the arrival sequence corresponds to a graph with many disconnected (often singletons) components and nodes with small degree. This causes PIVOT to perform better than the rest of the algorithms at the beginning of each arrival sequence as shown in Figure 6, but this behavior is not representative in a more natural arrival sequence (as in the case of cit-HepTh in Figure 2). To illustrate this, consider the case where the arrival sequence creates many small line-graphs of constant length. On each such component PIVOT includes a constant number of edges inside clusters with density more than half, hence the cost of the solution produced by PIVOT is a constant fraction smaller compared to SINGLETONS. On the other hand, AGREE-OFF will put all nodes of a line subgraph (of length > 2) into singleton clusters.

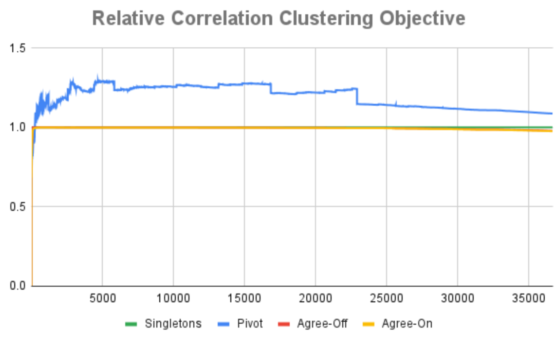


(a) Clustering cost, email-Enron. (b) Recourse, email-Enron, log-scale. (c) Run time, email-Enron, log-scale.

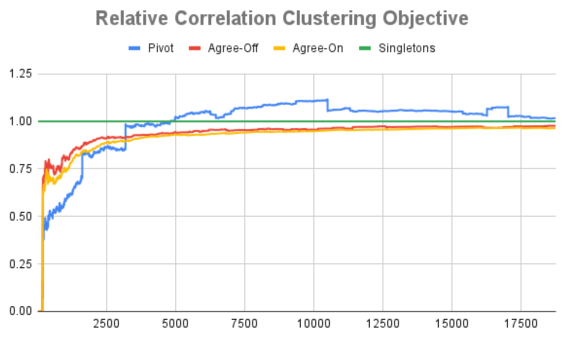
(d) Clustering cost, ca-AstroPh. (e) Recourse, ca-AstroPh, log-scale. (f) Run time, ca-AstroPh, log-scale.

(g) Clustering cost, musae-facebook (h) Recourse, musae-facebook, log-scale. (i) Run time, musae-facebook, log-scale.

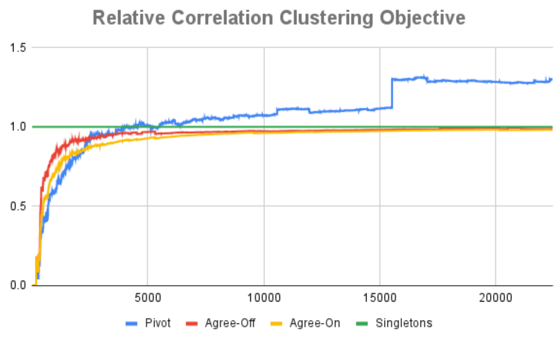
Figure 5. Comparison of our algorithm with the two baselines, for the datasets email-Enron, ca-AstroPh, and musae-facebook.



(a) Clustering cost relatively to the number of edges, email-Enron.



(b) Clustering cost relatively to the number of edges, ca-AstroPh.



(c) Clustering cost relatively to the number of edges, musae-facebook.

Figure 6. Comparison of our algorithm with the two baselines, for the datasets email-Enron, ca-AstroPh, and musae-facebook.