Sufficient Conditions for Edit-Optimal Clusters*

Peter Damaschke†
Department of Computer Science and Engineering
Chalmers University, 41296 Göteborg, Sweden
ptr@chalmers.se

Abstract

Cluster Editing is the problem of turning a graph into a cluster graph, that is, a disjoint union of cliques, by a minimum number of edge edits. Cluster Deletion is similarly defined, with edge deletions only. We propose a local notion of edit-optimality: Informally, we say that a crown (a certain type of labeled graph) is edit-optimal if it yields a cluster in some optimal solution to Cluster Editing, in *every* graph containing this crown as a subgraph. Then we give sufficient conditions for edit-optimality, e.g., in terms of vertex degrees. A condition for Cluster Deletion applies a theorem of Landau (1953) on degree sequences of tournaments. The conditions are particularly suited for planted models of clusterings, and for networks that only partially exhibit a clear cluster structure.

Keywords: graph algorithms, cluster editing, optimality criteria, degree sequence

1 Introduction

A cluster graph is a disjoint union of cliques. The Cluster Editing problem asks to turn a graph $G = (V, E)$ into a cluster graph by at most k edge edits, that is, insertions or deletions of edges. It has applications in the analysis of similarity data in, e.g., molecular biology [11] and has been intensively studied, see for instance [1, 2, 5, 9]. Cluster Editing is NP-hard [10] and fixed-parameter tractable (FPT) with parameter k, but also with alternative parameters, such as the edit degree (see definitions below) and number of clusters combined [6]. For the currently fastest FPT algorithms and smallest problem kernels we refer to [1, 4, 3].

Here we study the isolation of single subgraphs becoming clusters in optimal solutions to Cluster Editing. We call them edit-optimal. (We defer the somewhat technical exact definition.) As one particular motivation, large social networks tend to consist of a core without a clear cluster structure and a periphery with clusters [8]. To single out some optimal clusters is then more appropriate than an artificial cluster*ing* of the entire graph. Clusters can also be used for disambiguation of words in corpora of documents, and many other purposes in data mining. Efficient removal of optimal clusters can also serve as data reduction rules in

*In one section of a short conference paper at the 24th International Workshop on Combinatorial Algorithms IWOCA 2013, Lecture Notes in Computer Science (Springer), vol. 8288, pp. 433–437, the proposed concept was sketched and a few results announced, however without full proofs. This paper is an extended version with additional results.

†Tel. 0046 31 772 5405, Fax 0046 31 772 3663

Cluster Editing kernelizations as in [4, 3]). But here we study this matter for its own sake, due to the above motivation. We give sufficient conditions for edit-optimality, which are easy to verify in a graph and allow to conclude that Cluster Editing indeed yields the original cliques if these conditions are satisfied. Cluster Editing does in general *not* guarantee to reconstruct the right clusters. Furthermore, k can be large in practice, therefore it might also be pleasant to notice that our degree conditions relate to an alternative parameterization [6].

In Section 3 we introduce edit-optimal "crowns" that capture local properties of optimal clusters. In Section 4 we show that a maximum edit degree of at most 1/5 of the cluster size implies edit-optimality. Section 5 starts from the idea of packing conflict triples and gives a condition formulated in terms of special edge colorings. This interlude section is inserted because we need the result as a proof step in Section 6 which yields perhaps the most interesting result: We characterize edit-optimality in a case that naturally appears in cluster graphs with a few random edges added. Surprisingly, this is a new application of a classic combinatorial theorem of Landau from 1953 [7].

2 Preliminaries

We consider graphs $G = (V, E)$ with n vertices and m edges and use standard notation like $N(v) := \{u \in V : uv \in E\}$ for the open neighborhood of a vertex, $N[v] := N(v) \cup \{v\}$ for the closed neighborhood, and $N[W] := \bigcup_{v \in W} N[v]$ and $N(W) := N[W] \setminus W$ for neighborhoods of vertex sets. Let $G - W$ denote the subgraph of G induced on $V \setminus W$.

A *cluster graph* is a disjoint union of cliques. G is a cluster graph if and only if G has no *conflict triple*, i.e., three vertices inducing exactly two edges. (Trivially, a disjoint union of cliques cannot contain conflict triples. Conversely, in the absence of conflct triples, the relation "$u = v$ or u, v are adjacent" is transitive, thus an equivalence relation, and its equivalence classes are cliques.)

The Cluster Editing problem takes as input a graph $G = (V, E)$, and asks to turn it into a cluster graph by at most k edge edits, that is, insertions or deletions of edges. We also call an edited edge simply an *edit*. Given a solution, the *edit degree* of a vertex v is the number of edits incident to v. The term *clustering* refers in this paper to any partitioning of the vertex set V (without special properties required), and the subsets of the partitioning are called *clusters*. However, an *optimal clustering* means an optimal solution to Cluster Editing.

In Cluster Editing, all edits have unit costs. In the *weighted* version of Cluster Editing, every pair of vertices has some non-negative cost, and the total cost of a solution is the sum of weights of all edits. A special case is Cluster Deletion where only edge deletions are permitted, in other words, insertions have infinite costs.

A *star* is a graph with one *center* vertex adjacent to *leaf* vertices of degree 1.

A directed graph D with c vertices is a *simple digraph* if it has neither self-loops nor multiple (parallel or opposite) edges between any two vertices. The *score sequence* is the nondecreasing sequence of the outdegrees $s_1 \leq \ldots \leq s_c$, and D is called a *tournament* if $\sum_{i=1}^{c} s_i = \binom{c}{2}$.

3 A Notion of Edit-Optimality

Isolating a set $C \subseteq V$ in a graph $G = (V, E)$ means the following operation: Delete all edges between C and $V \setminus C$, called *cut edges*, and insert all missing edges in C to make it a clique. Moreover, we declare C a cluster in a Cluster Editing solution. The edit degree of any $v \in C$ is then the number of edits incident to v caused by isolating C.

Suppose we want to solve Cluster Editing on G. Once we have recognized (by any criterion) the *existence* of some optimal clustering having C as a cluster, we can safely isolate C and then solve Cluster Editing on $G - C$. We cannot expect to find such clusters C in polynomial time, because iterated application of this step would solve Cluster Editing in polynomial time. Therefore we only quest for *sufficient* conditions π of the form: If some set C satisfies π then some optimal clustering exists with C as one of its clusters. Ideally, π should be easy to apply and not too restrictive.

Whether or not C is a cluster in some optimal clustering cannot be decided only by the structure of the subgraph induced by C. The issue is illustrated by a simple example: Let $G := P_n$, the chordless path of n vertices. Let $C := \{v\}$ where v is one end of the path. If n is odd, then an optimal clustering is obtained by isolating C and splitting the remaining P_{n-1} into pairs of P_2, by $(n-1)/2$ edits. This is optimal since P_n has $(n-1)/2$ conflict triples that share no vertex pairs, but all conflict triples must be edited. But if n is even, then isolating C costs 1 edit and leaves us with P_{n-1}, $n-1$ odd. Due to the previous case we need $1 + (n-2)/2 = n/2$ edits, whereas splitting P_n into P_2 costs only $n/2 - 1$ edits. Hence it depends on the entire graph whether C appears as a cluster in some optimal clustering. It would be useful to have sufficient conditions π that rely only on C and its surrounding. It is natural to ask "how local" such properties π can be. First we observe that even sufficient conditions π cannot rely only on the subgraph induced by C, since there always exists a graph $G = (V, E)$, with any given induced subgraph on $C \subset V$, where C fails to be a cluster in any optimal optimal clustering. For instance, let $N(C)$ be a huge clique, and let all possible edges between C and $N(C)$ exist. – Thus it is not enough to restrict attention to C only. A "minimal extension" is now to look also at the incident edges that leave C. More formally, we fist define two auxiliary graphs as follows.

Definition 1. *Let $C \subseteq V$ be a subset of vertices in $G = (V, E)$. We define two vertex-labeled graphs $H^*(C)$ and $H(C)$ on the vertex set $N[C]$ as follows: The edges of $H^*(C)$ are the edges (of G) in C, and the cut edges between C and $N(C)$. The edges of $H(C)$ are the non-edges in C and the cut edges between C and $N(C)$. Let $I := N(C)$, note that $N[C] = C \cup I$. The label of each vertex $v \in N[C]$ tells whether $v \in C$ or $v \in I$. We say that the set $C \subseteq V$ in $G = (V, E)$* generates *the labeled graph $H^*(C)$. Let $m(H(C))$ denote the number of edges of $H(C)$.*

Note that $H^*(C)$ is some (non-induced) subgraph of G. Graph $H(C)$ differs from $H^*(C)$ only in that we take the complement of the subgraph of G induced on C. In other words, the edges of $H(C)$ are now exactly the edits incident to C when we isolate C. We defined both versions because this will make some formulations more convenient later on. The next definition differs from Definition 1 in that it does not refer to the graph G that hosts H and H^*. (The name "crown" is inspired by similar graph concepts used elsewhere and must not be mixed up with them.)

Definition 2. *A* crown H^* *is a vertex-labeled graph with vertex set* $C \cup I$*, such that:* $C \cap I = \emptyset$*,* $N(C) = I$*, and* I *is an independent set. The label of each vertex* $v \in N[C]$ *tells whether* $v \in C$ *or* $v \in I$*. Let* H *be the crown obtained from* H^* *by replacing all edges in* C *with non-edges and vice versa. Let* m *be the number of edges of* H*.*

The connection to Cluster Editing is now made by:

Definition 3. *Consider the weighted* Cluster Editing *problem on* H^**, where edges in* I *may be inserted for free, while edits incident to* C *have unit cost. We call the crown* H edit-optimal *if every solution to weighted* Cluster Editing *on* H^* *has total costs of at least* m*.*

Proposition 4. *A crown* H *is edit-optimal if and only if: For every graph* G *and every vertex set* C *such that* $H(C)$ *is isomorphic to* H*, the set* C *is a cluster in some optimal clustering of* G *(in order words: it is safe to isolate* C *when solving* Cluster Editing *on* G*).*

Proof. Let C be any subset of the vertex set of G. The number of edits incurred by any clustering of G is the sum of two terms: the number $e(C)$ of edits incident to C, and the number of edits within $G - C$. Furthermore, $e(C) = m(H(C))$ only depends on the clustering induced on $N[C]$.

Consider any clustering of G and denote the clusters $C_1, \ldots, C_d$. If we isolate C and also replace every cluster C_i with $C_i \setminus C$, then the number of edits in $G - C$ is not changed, and $e(C) = m(H(C))$.

Hence, whenever $e(C) \geq m(H(C))$ in the original clustering, then isolation of C does not increase the number of edits. It follows: *If every clustering of* $H^*(C)$ *requires at least* $m(H(C))$ *edits incident to* C, then it is optimal to isolate C. We can equivalently express the highlighted condition by virtue of weighted Cluster Editing of $H^*(C)$, where an edit incident to C costs 1, and edits within $N(C)$ cost 0: If weighted Cluster Editing of $H^*(C)$, where edges in $N(C)$ can be inserted for free, costs at least $m(H(C))$, then it is optimal to isolate any set that generates a labeled graph isomorphic to $H^*(C)$, and this holds independently of the rest of the graph.

The converse is also true: If the highlighted condition is violated, then there exists a graph G with a set C generating $H^*(C)$, where it is not optimal to isolate C. Namely, we can obtain one such G as follows. The vertex set of our G is only $N[C]$. By assumption, weighted Cluster Editing on $H^*(C)$ has a solution with fewer than $m(H(C))$ edits incident to C. We take such a solution and insert in $I = N(C)$ the edges edited there (for free), in addition to the edges from $H^*(C)$. Note that *Cluster Editing* on this graph G now has a solution with cost less than $m(H(C))$, whereas isolating C would cost $m(H(C))$.

Together with Definition 3, the following statements are thus equivalent: H is an edit-optimal crown; weighted Cluster Editing on H^* costs at least m; isolating C costs at least $m(H(C))$ in every graph where $H(C)$ is isomorphic to H; isolating C is optimal in every graph where $H(C)$ is isomorphic to H. □

That is, the technically looking Definition 3 gives a condition π as we aimed for. The following sections provide some sufficient conditions for edit-optimal crowns that are easy to verify in a given instance.

4 A Sufficient Condition in Terms of Edit Degrees

Theorem 5. *Let $C \subseteq V$ be a subset of vertices in a graph $G = (V, E)$, such that $H(C)$ has maximum degree t, and $c := |C| \geq 5t$. Then C is a cluster in some optimal clustering of G. Consequently, every crown H on a vertex set $C \cup I$, with vertex degrees at most $|C|/5$, is edit-optimal.*

Proof. Consider any clustering and denote the clusters $C_1, \ldots, C_d$. We study what happens when we isolate C, that is, make C a new cluster and replace every C_i with $C_i \setminus C$. Some of these differences may become empty dummy clusters, but empty clusters obviously do not affect the number of edits, thus we need not pay special attention to them. A pair uv of vertices stops being an edit in exactly the following cases:
(1) uv is a non-edge with $u \in C \cap C_i$ and $v \in C_i \setminus C$ for some i.
(2) uv is an edge with $u \in C \cap C_i$ and $v \in C \cap C_j$ for some $i \neq j$.
In cases (1) and (2) we call uv a good pair. Similarly, a pair uv of vertices becomes a new edit in exactly the following cases:
(1') uv is an edge with $u \in C \cap C_i$ and $v \in C_i \setminus C$ for some i.
(2') uv is a non-edge with $u \in C \cap C_i$ and $v \in C \cap C_j$ for some $i \neq j$.
In cases (1') and (2') we call uv a bad pair. Clearly, the number of edits does not increase if every bad pair is compensated by at least one good pair.

In the following it is convenient to think of every pair uv as a stick that can be broken in two halves, namely u's end and v's end of uv. Since the edit degrees in $N[C]$ are bounded by $0.2c$, every vertex in $N[C]$ is involved in at most $0.2c$ new edits, i.e., bad pairs. Consider any index i where $|C \cap C_i| \leq 0.4c$, and any vertex $u \in C \cap C_i$. At least $0.6c$ pairs uw, $w \in C \setminus C_i$, exist. We compensate every bad pair uv with u's sides of two good pairs uw. Since u is in at most $0.2c$ bad pairs, and $0.2 + 2 \cdot 0.2 = 0.6$, we see that enough good pairs uw are available for that. Next, consider i where $|C \cap C_i| > 0.4c$; of course, at most two such indices i can exist. Bad pairs of type (1'), that is, $u \in C \cap C_i$ and $v \in C_i \setminus C$, are compensated as follows. Every v is in at most $0.2c$ bad pairs, hence more than $0.2c$ pairs of v with vertices of $C \cap C_i$ are good, and we use them for compensation. Bad pairs of type (2') are already compensated, except possibly those between two sets $C \cap C_i$ and $C \cap C_j$ of size larger than $0.4c$. But again, every $u \in C \cap C_i$ is in at most $0.2c$ bad pairs uv with $v \in C \cap C_j$, hence more than $0.2c$ other pairs uv are good and can be used for compensation. Now all bad pairs are compensated (if not overcompensated), and no end of a good pair is used twice. □

By Theorem 5, some bounded edit degree $t = O(c)$ implies edit-optimality, whereas the total number of edits incident to C can be $tc = O(c^2)$. Thus Theorem 5 requires much less than, e.g., the condition $2\delta(C) + \gamma(C) < c$ from [3], where $\delta(C)$ and $\gamma(C)$ denote the number of non-edges in C and cut edges between C and $G - C$, respectively.

5 A Sufficient Condition in Terms of List Edge Colorings

An *edge coloring* of a graph $G = (V, E)$ is a function $p : E \longrightarrow Col$ from the edge set E into a set Col of colors, such that $p(e) \neq p(f)$ for any two incident edges e and f. Now assume that every edge e has its individual list $S(e) \subseteq Col$ of suitable colors. A *list edge coloring* is an edge coloring p such that $p(e) \in S(e)$ holds for every $e \in E$.

In the following we derive another sufficient conditions for edit-optimality, in terms of list edge colorings with a simple additional constraint. Let H be a crown with vertex set $C \cup I$ and edge set E. Let us think of the vertices of C as colors, that is, let $Col := C$ from now on. We say that an edge coloring $p : E \longrightarrow Col$ of H has *mutual usage of colors*, if two edges e and f exist, such that $p(e)$ is some vertex of f, and $p(f)$ is some vertex of e.

Proposition 6. *let H be a crown with vertex set $C \cup I$, let C also be the set of colors, and for every edge uv of H, let $S(uv)$ be the set of all $s \in C$ that are adjacent neither to u nor to v in H. Now, if H admits a list edge coloring without mutual usage of colors, then H is edit-optimal.*

Proof. Informally, the basic idea is that a packing of conflict triples gives a lower bound on the edit costs. In detail:

Let m be the number of edges in H. Suppose that H has a list edge coloring p, as specified above, without mutual usage of colors. We only have to show that every solution to weighted Cluster Editing on H^*, where edge insertions in I are for free, has a total cost at least m (see Definition 2 and 3).

Consider each of the m edges uv and let $s := p(uv) \in S(uv)$. We claim that u, v, s form a conflict triple in H^*. Simply check the two cases: If $u, v \in C$ then (u, s, v) is a path and uv is not an edge in H^*. If $u \in C$ and $v \in I$ (the opposite case is symmetric) then (s, u, v) is a path and uv is not an edge in H^*. Next we claim that no two of the so obtained conflict triples, say from edges e and f in H, share two vertices. Assume they do. If e and f are disjoint, then this is possible only if $p(e)$ is some vertex of f, and $p(f)$ is some vertex of e, which is mutual usage of colors. Thus e and f are incident, say $e = uw$ and $f = vw$, then by the definition of $S(uw)$ and $S(vw)$ the "color vertices" $p(e)$ and $p(f)$ can be none of the vertices u, w, v. But this implies $p(e) = p(f)$, hence p is not an edge coloring.

Thus we have found m conflict triples with the following properties: they pairwise share at most one vertex (i.e., no pair of vertices), and at least two vertices of each conflict triple are in C. Therefore, in every clustering of H^*, every such conflict triple contributes at least 1 to the edit costs. □

Proposition 6 can be applied directly, however the main reason for featuring it is that we need it in the following section. It is devoted to an interesting case where we are able to characterize and efficiently check edit-optimality.

6 Unions of Stars

Suppose that the crown H is a disjoint union of stars with centers in C. (Recall the definitions in Section 2.) That is, H has no edges in C, and all vertices of I have degree 1 in H. This case appears in planted models where the hidden clustering has many small clusters, and the errors have low probability and are one-sided, in the sense that additional edges are inserted between the clusters which are still cliques. We may ask under which circumstances Cluster Deletion correctly recovers the clustering. Under the given assumptions, for every fixed cluster C it is unlikely that any two extra edges incident to C share an end vertex, hence $H(C)$ is likely to be a disjoint union of stars. Therefore it is a relevant question what are the edit-optimal crowns H of this special type. Luckily we can completely and efficiently characterize them. As one prerequisite we use Landau's theorem [7]:

Theorem 7. *A sequence of nonnegative integers $s_1 \leq \ldots \leq s_c$ is the score sequence of some tournament on c vertices if and only if:*

$$\forall j \in \{1, \ldots, c-1\} : \sum_{i=1}^{j} s_i \geq \binom{j}{2} \ \wedge \ \sum_{i=1}^{c} s_i = \binom{c}{2}.$$

The inequalities in Theorem 7 can be rewritten as constraints on the suffix sums:

$$\forall j \in \{1, \ldots, c-1\} : \sum_{i=j+1}^{c} s_i \leq j(c-j) + \binom{c-j}{2}$$

Now we can also characterize the score sequences of simple digraphs in general:

Theorem 8. *A sequence of nonnegative integers $s_1 \leq \ldots \leq s_c$ is the score sequence of some simple digraph if and only if, for all $j = 0, \ldots, c-1$, we have*

$$\sum_{i=j+1}^{c} s_i \leq j(c-j) + \binom{c-j}{2}.$$

Proof. Suppose that a simple digraph with the given score sequence exists. We insert directed edges arbitrarily, to obtain a tournament, and we sort the new score sequence (since the new scores may no longer be nondecreasing). The suffix sums of the so obtained score sequence of the tournament, for $j = 1, \ldots, c-1$, satisfy the inequalities of Landau's theorem. If we undo the sorting and recover the original order of vertices, the inequalities remain valid, since the suffix sums cannot increase. Finally, the original scores are no larger, hence they satisfy the claimed inequalities for $j > 0$. The inequality for $j = 0$ is obviously true.

Conversely, we show that any score sequence with the given suffix sum inequalities can be realized by a simple digraph. Our proof strategy is to raise some scores while preserving the inequalities, until their sum is $\binom{c}{2}$. Then the score sequence meets the criteria of Landau's theorem, hence there exists a tournament. By choosing, for every vertex i, s_i outgoing edges arbitrarily, we get the desired digraph. It remains to show that, as long as

$$\sum_{i=1}^{c} s_i < \binom{c}{2},$$

we can always add 1 to some s_i and preserve all inequalities. Let k be minimal with

$$\sum_{i=k+1}^{c} s_i = k(c-k) + \binom{c-k}{2},$$

that is, the suffix sum inequalities are strict for all $j < k$. If they are all strict, we formally define $k := c$. Here we use the convention that a sum is zero if the lower index is larger than the upper index, e.g., $\sum_{i=c+1}^{c} s_i = 0$. Also note that $k > 0$, since otherwise we have already $\binom{c}{2}$ edges. Incrementing some s_j, $j \leq k$, by 1 leaves the inequalities for all $j < k$ valid (as they were strict), and trivially, it does not change the later suffix sums, for $j \geq k$. The only subtlety is that the selected j should satisfy $s_j + 1 \leq s_{j+1}$, such that we need not change the nondecreasing order of scores. Clearly, we always find such j, unless $k < c$ and

$s_1 = \ldots = s_k = s_{k+1}$. If $k = c$, we can raise s_c. For $k < c$ we show now that $s_k = s_{k+1}$ is impossible, which then concludes the proof. Observe that:

$$\sum_{i=k+1}^{c} s_i = k(c-k) + \binom{c-k}{2}.$$

$$\sum_{i=k+2}^{c} s_i \leq (k+1)(c-k-1) + \binom{c-k-1}{2}.$$

Their difference is

$$s_{k+1} = \sum_{i=k+1}^{c} s_i - \sum_{i=k+2}^{c} s_i \geq k(c-k) + \binom{c-k}{2} - (k+1)(c-k-1) - \binom{c-k-1}{2} = k.$$

Now, $s_k = s_{k+1} \geq k$ would imply

$$\sum_{i=k}^{c} s_i = s_k + \sum_{i=k+1}^{c} s_i \geq k + k(c-k) + \binom{c-k}{2}$$

$$= (k-1) + k(c-k) + \binom{c-k}{2} + 1 = (k-1)(c-k+1) + \binom{c-k+1}{2} + 1,$$

contradicting the assumed suffix sum inequality for $j = k-1$. □

Lemma 9. *Let the crown H be a union of stars, where the center vertices in C have degrees $s_1 \leq \ldots \leq s_c$. Then, H admits a list edge coloring without mutual usage of colors, if and only if the center degrees form the score sequence of some simple digraph.*

Proof. Given a list edge coloring without mutual usage of colors, we insert a directed edge us in C, for every edge uv ($u \in C$, $v \in I$) of H colored by $s \in C$. Since incident edges have distinct colors, we do not generate parallel directed edges. Since no two edges of H mutually use each other's vertices as colors, we do not generate opposite directed edges either. Hence we get a simple digraph on C with the prescribed scores.

Suppose that $s_1 \leq \ldots \leq s_c$ form the score sequence of a simple digraph on vertex set C. Specifically, imagine that s_i outgoing directed edges are attached to vertex i with degree s_i in H, for $i = 1, \ldots, c$. Then we color the s_i edges of the star with center i with the end vertices of these directed edges. Correctness arguments are converse to the previous ones. □

Theorem 10. *Let H be a union of stars, where the centers in C have degrees $s_1 \leq \ldots \leq s_c$. Then, H is edit-optimal if and only if the center degrees form the score sequence of some simple digraph. Moreover, this condition can be trivially checked in polynomial time.*

Proof. The "if"-direction follows from Proposition 6 and Lemma 9. Conversely, suppose that H is edit-optimal. According to Definition 2 this means that every clustering of H^* costs at least $m = \sum_{i=1}^{c} s_i$. In particular, for every fixed $j = 0, \ldots, c-1$ we consider the following clustering: Every star with center $j+1, \ldots, c$ is a cluster of its own, the rest of C with centers indexed at most j is one cluster, and the leaves of those centers form another cluster. Remembering how the unit and zero costs of edits in H^* are defined, we see that the total cost equals

$$\sum_{i=1}^{j} s_i + j(c-j) + \binom{c-j}{2} \geq m = \sum_{i=1}^{c} s_i.$$

Hence we have

$$\sum_{i=j+1}^{c} s_i \leq j(c-j) + \binom{c-j}{2}.$$

Due to Theorem 8 this is exactly the criterion for a score sequence of a simple digraph. □

Acknowledgment

This work has been supported by the Swedish Foundation for Strategic Research (SSF) through Grant IIS11-0089 for a data mining project entitled "Data-driven secure business intelligence".

References

[1] S. Böcker, A golden ratio parameterized algorithm for cluster editing, J. Discr. Algor. 16 (2012) 79–89

[2] S. Böcker, S. Briesemeister, G.W. Klau, Exact algorithms for cluster editing: Evaluation and experiments, Algorithmica 60 (2011) 316–334

[3] Y. Cao, J. Chen, Cluster editing: Kernelization based on edge cuts, Algorithmica 64 (2012) 152–169

[4] J. Chen, J. Meng, A $2k$ kernel for the cluster editing problem, J. Comput. System Sci. 78 (2012) 211–220

[5] F.V. Fomin, S. Kratsch, M. Pilipczuk, M. Pilipczuk, Y. Villanger, Tight bounds for parameterized complexity of cluster editing, in: N. Portier, T. Wilke (Eds.) STACS 2013. LIPIcs, vol. 20, Dagstuhl 2013, pp. 32–43

[6] C. Komusiewicz, J. Uhlmann, Cluster editing with locally bounded modifications, Discr. Appl. Math. 160 (2012) 2259–2270

[7] H.G. Landau, On dominance relations and the structure of animal societies. III. The condition for a score structure, Bull. Math. Biophysics 15 (1953) 143–148

[8] J. Leskovec, K.J. Lang, A. Dasgupta, M.W. Mahoney, Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters, Internet Math. 6 (2009) 29–123

[9] S. Rahmann, T. Wittkop, J. Baumbach, M. Martin, A. Truß, S. Böcker, Exact and heuristic algorithms for weighted cluster editing, in: P. Markstein, Y., Xu (Eds.) CSB 2007, Imperial College Press 2007, pp. 391–401

[10] R. Shamir, R. Sharan, D. Tsur, Cluster graph modification problems, Discr. Appl. Math. 144 (2004) 173–182

[11] R. Sharan, R. Shamir, Algorithmic approaches to clustering gene expression data, in: Current Topics in Computational Molecular Biology, MIT Press 2002, pp. 269–300