Bounded-Degree Spanning Trees

Problem Statement

Let $G = (V, E)$ be a connected undirected graph, and for each vertex $v \in V$ let $b_v \geq 0$ be an integer upper bound on its degree. The objective is to find a spanning tree T of G minimizing the maximum degree exceedance

$$\max_{v \in V}(d_T(v) - b_v),$$

where $d_T(v)$ denotes the degree of v in T.

Result

We present a local-search algorithm that either finds a feasible tree satisfying all bounds (i.e., $d_T(v) \leq b_v$ for all v), or else returns a tree whose maximum exceedance is at most one more than the optimal possible exceedance. Formally,

$$\max_{v \in V}(d_T(v) - b_v) \;\leq\; \mathrm{OPT}_{\mathrm{exc}} + 1,$$

where

$$\mathrm{OPT}_{\mathrm{exc}} = \min_{T' \in \text{ spanning tree}} \max_{v \in V}(d_{T'}(v) - b_v).$$

1 Setup and notation

- For a spanning tree T and vertex v define the *exceedance*

$$e_T(v) := d_T(v) - b_v.$$

- For a fixed tree T let

$$k := \max_{v \in V} e_T(v) \qquad (\text{so } k \geq 0).$$

- Define also

$$D_k := \{v \in V : e_T(v) = k\}$$

$$D_{k-1} := \{v \in V : e_T(v) = k - 1\}$$

$$S := D_k \cup D_{k-1}$$

  Note that D_k and D_{k-1} are disjoint, so $|S| = |D_k| + |D_{k-1}|$.

- Let F denote the set of edges of T that are incident to at least one vertex in S. Removing the $|F|$ edges of F from T splits the tree into exactly $|F| + 1$ connected components; call this collection of components $\mathcal{C}$.

2 Algorithm (phases and subphases)

Below is the local-search algorithm that implements the phased/subphased approach (the +1 guaranty algorithm). This section fits into the notation above: phases operate at the current exceedance level k, and subphases remove one exceedance-k vertex at a time while preserving the invariant that no new exceedance-k vertices are created.

Informal overview

- A *phase* fixes the current maximum exceedance k (computed from T). The goal of the phase is to eliminate all vertices with exceedance k.
- A *subphase* reduces exactly one vertex from exceedance k to $k-1$ while never creating a new exceedance-k vertex. Each phase consists of multiple subphases; after all exceedance-k vertices are removed, the phase ends and a new phase starts with a possibly lower k.
- During a subphase, we consider the set $S = D_k \cup D_{k-1}$, its incident tree-edge set F, and the components $\mathcal{C}$ of $T \setminus F$. We scan non-tree edges that connect different components of $\mathcal{C}$. If every such non-tree edge touches S, Lemma 1 gives the certificate that $\mathrm{OPT}_{\mathrm{exc}} \geq k-1$ and hence $\Delta(T) \leq \mathrm{OPT}+1$, so we can terminate. Otherwise we find a non-tree edge whose endpoints are outside S; adding it creates a cycle in $T \cup \{e\}$ which we use to either (a) mark exceedance-$(k-1)$ vertices as reducible (if the cycle contains no degree-k vertices), or (b) perform a local swap to reduce some exceedance-k vertex (if the cycle contains a exceedance-k vertex). Marked reducible vertices are executed in a propagation sequence when needed so that no endpoint's exceedance ever reaches k.

Precise definitions used in a subphase

- $D_k = \{v : e_T(v) = k\}$ and $D_{k-1} = \{v : e_T(v) = k-1\}$.
- $S = D_k \cup D_{k-1}$.
- F is the set of tree edges incident to S.
- $\mathcal{C}$ is the collection of connected components of $T - F$.
- A non-tree edge $(v, w) \in E \setminus T$ *crosses* $\mathcal{C}$ if v and w lie in different components of $\mathcal{C}$.
- If adding a non-tree edge (v, w) forms a cycle C in $T \cup \{(v, w)\}$ with $C \cap D_k = \varnothing$, then every $u \in C \cap D_{k-1}$ is *marked reducible* via (v, w) (we record the corresponding cycle). Marked vertices are reduced later in a propagation order when required.

Propagation claim

Marked reducible vertices are executed in the order they were discovered (or in an order determined by dependency). The key propagation claim is:

> If a vertex $x \in D_{k-1}$ is marked reducible by some cycle discovered earlier in the subphase, then the recorded swap that reduces x can be carried out later without causing any vertex's exceedance to reach k (provided earlier recorded moves are executed first).

The usual inductive argument applies: when a reducible vertex is recorded its cycle lies entirely inside components of $\mathcal{C}$ free of exceedance-k vertices; reducing previously recorded vertices only lowers degrees/exceedances further, so executing earlier recorded moves cannot create new exceedance-k vertices and thus the swap for x is safe when reached.

Termination and correctness of the phased algorithm

- Each subphase reduces the quantity $|D_k|$ by at least one and never increases it (by the propagation claim). Hence each phase performs at most $|V|$ subphases.
- If at some point every crossing non-tree edge touches S then Lemma 1 implies $\mathrm{OPT}_{\mathrm{exc}} \geq k-1$, so current $k \leq \mathrm{OPT}_{\mathrm{exc}} + 1$ and the algorithm may terminate with the +1 guarantee.

Pseudocode (one phase / one subphase)

Algorithm 1 Local-search with phases & subphases

```
Let T be any spanning tree of G.
loop                                                         ▷ Start a new phase
    k ← max_v e_T(v)
    while D_k ≠ ∅ do                                         ▷ Start a new subphase
        Compute D_k, D_{k−1}, S = D_k ∪ D_{k−1}.
        Let F be tree edges incident to S, and let 𝒞 be components of T \ F.
        Mark all vertices in D_{k−1} as unlabeled.
        if every non-tree edge that crosses 𝒞 has an endpoint in S then
            return T                                         ▷ by Lemma 1, OPT_exc ≥ k − 1.
        end if
        for all non-tree edges (v, w) crossing 𝒞 with v, w ∉ S do
            Let C be the unique cycle in T ∪ {(v, w)}.
            if C ∩ D_k = ∅ then
                For each u ∈ C ∩ D_{k−1}: mark u reducible via (v, w).
                Remove marked vertices from D_{k−1} and update F, 𝒞 accordingly.
            else
                Pick some u ∈ C ∩ D_k.
                if either endpoint v or w is marked reducible then
                    Execute the recorded reducible move(s) (propagate as needed) so v, w are safe.
                end if
                Execute local swap: add (v, w) to T and remove a u–incident tree edge on C.
                                                             ▷ This reduces u from exceedance k to k − 1
                Break for loop                               ▷ Start new subphase
```

3 Key lemma (lower bound on OPT)

Lemma 1. *Assume a local-reduction move is not available. Then for the current tree T with exceedance k we have*

$$\mathrm{OPT}_{\mathrm{exc}} \;\geq\; k-1.$$

Equivalently, if no local move is possible at level k, then any spanning tree T' must have some vertex of exceedance at least $k-1$.

Proof. We follow the combinatorial counting argument.

(1) Any spanning tree needs many incidences into S. The removal of the $|F|$ edges incident to S produced $|F|+1$ components (the set $\mathcal{C}$). To reconnect these components into a spanning tree one needs at least $|F|$ edges whose endpoints lie in different components of $\mathcal{C}$. By the assumption that no local-reduction move exists at level k, every edge that connects two distinct components of $\mathcal{C}$ must have at least one endpoint in S. Therefore, for any spanning tree T' we have that the total degree (incidence count) of vertices of S in T' is at least $|F|$:

$$\sum_{v\in S} d_{T'}(v) \;\geq\; |F|. \tag{1}$$

(2) Convert to exceedances. Subtract the bounds b_v from both sides of (1) to obtain

$$\sum_{v\in S} \big(d_{T'}(v) - b_v\big) \;\geq\; |F| - \sum_{v\in S} b_v.$$

Dividing by $|S|$ shows that the average exceedance of vertices in S in any spanning tree T' satisfies

$$\frac{1}{|S|}\sum_{v\in S} e_{T'}(v) \;\geq\; \frac{|F| - \sum_{v\in S} b_v}{|S|}. \tag{2}$$

Since the maximum exceedance of T' is at least its average exceedance on any nonempty subset, $\mathrm{OPT}_{\mathrm{exc}}$ is bounded below by the RHS of (2).

(3) Lower bound $|F|$ using the current tree T. In the current tree T the degree of each vertex in D_k equals $b_v + k$, and the degree of each vertex in D_{k-1} equals $b_v + k - 1$. Therefore the sum of degrees of vertices in S, taken inside T, is

$$\sum_{v\in D_k} (b_v + k) \;+\; \sum_{v\in D_{k-1}} (b_v + k - 1) \;=\; \sum_{v\in S} b_v \;+\; k|D_k| + (k-1)|D_{k-1}|.$$

This sum counts each edge of T with both endpoints in S twice; edges with exactly one endpoint in S are counted once. The number of edges inside S is at most $|S| - 1$ because these edges form a forest on the vertex set S (they are a subgraph of the tree T). Thus the number of edges incident to S (which is $|F|$) satisfies

$$|F| \;\geq\; \sum_{v\in S} b_v \;+\; k|D_k| + (k-1)|D_{k-1}| \;-\; (|S| - 1). \tag{3}$$

A small algebraic rearrangement yields

$$|F| - \sum_{v\in S} b_v \;\geq\; (k-1)|S| - |D_{k-1}| + 1. \tag{4}$$

(4) Combine (2) **and** (4)**.** Substituting (4) into (2) gives

$$\frac{1}{|S|}\sum_{v\in S} e_{T'}(v) \;\geq\; \frac{(k-1)|S| - |D_{k-1}| + 1}{|S|} \;=\; (k-1) - \frac{|D_{k-1}| - 1}{|S|}.$$

Since $|D_{k-1}| \leq |S| - 1$ (because D_k is nonempty), we have $0 \leq \frac{|D_{k-1}|-1}{|S|} < 1$. Therefore

$$\frac{1}{|S|}\sum_{v\in S} e_{T'}(v) \;>\; k-2 \qquad \text{and} \qquad \frac{1}{|S|}\sum_{v\in S} e_{T'}(v) \;\geq\; k - 2 + 1 = k - 1.$$

More importantly, the quantity on the left has ceiling at least $k-1$, which implies that the maximum exceedance in any spanning tree T' is at least $k-1$. In symbols,

$$\mathrm{OPT}_{\mathrm{exc}} \;\geq\; k-1.$$

This proves the lemma, giving the desired $+1$ guarantee.

$\square$