1 Skeleton dimension

A graph is $\gamma$-doubling if any ball of radius $r$ is covered by $\gamma$ balls of radius $r/2$. More precisely, for any node $u$ and radius $r > 0$, there exist nodes $u_1, \ldots, u_k$ with $k \leq \gamma$ such that $B(u, r)$ is included in $\bigcup_{i=1}^{k} B(u_i, r/2)$. Show that a graph with skeleton dimension $k$ is $2k + 1$-doubling.

2 Hopsets

Given a connected weighted undirected graph $G$, we define the $h$-hop distance $d^h_{G}(u,v)$ between $u$ and $v$ as the minimum weight of a path from $u$ to $v$ with $h$ edges at most. (Each edge of a path is called a “hop”, it thus corresponds to the distance using at most $h$ hops.) The usual distance from $u$ to $v$ is thus $d_{G}(u,v) = d^{|V(G)| - 1}_{G}(u,v)$. We define a $h$-hopset of $G$ as a set $H$ of edges such that $d^h_{G,H}(u,v) = d_{G}(u,v)$ where each edge $uv$ of $H$ is considered to have weight $d_{G}(u,v)$.

a) What notion seen in course is tightly related to the notion of 2-hopset?
b) Suppose that $G$ is a path of length $n$, propose a 2-hopset of $G$ with as few edges as you can (we do not care about multiplicative constants).
c) Same question for a 3-hopset.
d) Same question for a 4-hopset.

3 Hierarchical hub labeling from contraction hierarchies

Given an unweighted undirected graph $G$ and a contraction order $\pi = u_1 \prec \cdots \prec u_n$ of the nodes of $G$, we consider the contraction hierarchies $E^+ = E^+_n$ of directed edges where $E^+_0 := \{vw \in E(G) \mid v \prec w\}$ and $E^+_i := E^+_{i-1} \cup \{vw \mid u_i \prec v < w \text{ and } u_iw \in E^+_{i-1}\}$. We define the hub-set $H_u$ of $u$ as the set of nodes visited by a BFS from $u$ in the directed graph $G^+_u = (V(G), E^+)$. We give the main argument for proving that $(H_u)_{u \in V(G)}$ has the following covering property: for all pair $u, v \in V(G)$ and for any shortest path $P$ between $u$ and $v$, we have $H_u \cap H_v \cap P \neq \emptyset$. 
b) Recall how the distance between $u$ and $v$ can be computed from $H_u$ and $H_v$ if appropriate information is associated to elements of $H_u$ and $H_v$.

c) Give the main argument for proving that $(H_u)_{u \in V(G)}$ is hierarchical, i.e. the relation “is hub of is transitive: for all triple $u, v, w \in V(G)$, if $v \in H_u$ and $w \in H_v$, then $w \in H_u$.

Solution:

a) The node $u_i \in P$ with $i$ maximal is visited by both BFS.

b) Associate $d(u, a)$ to $a \in H_u$; $d(u, v)$ can be obtained as $\min_{a \in H_u \cap H_v} d(u, a) + d(v, a)$ according to a) and triangle inequality.

c) If $v$ is visited by the BFS from $u$, then all edges considered in a BFS from $v$ will be (or where already) considered in the BFS from $u$, implying $H_v \subseteq H_u$.

4 Connection Scan Algorithm Revisited

We consider a bus network given by an array $C$ of connections. More precisely, each connection $c \in C$ represents the elementary travel of a bus departing from a stop $c$.from at time $c$.dep and arriving at the next stop $c$.to at time $c.arr > c.dep$. $C$ is sorted by increasing departure time. For simplicity, we assume that all departure times are distinct ($c.dep \neq c'.dep$ for $c \neq c'$). A sequence $c_1, \ldots, c_k$ of connections is a feasible journey if $c_i$.to $= c_{i+1}.from$ and $c_i.arr \leq c_{i+1}.dep$ for all $i = 1 \ldots k - 1$ (no walk is considered). When a traveler starts at a given stop $src$, we store at every stop $u$ an estimation $\tau(u)$ of arrival time at $u$. (Initially, $\tau(src)$ stores the starting time at $src$ and $\tau(u) = \infty$ for $u \neq src$.)

a) Earliest arrival: Propose a linear time algorithm for computing the earliest arrival time at a given stop $dst$.

b) Last departure: Propose a linear time algorithm for computing the last departure time from $src$ such that $dst$ can be reached at a given time $\tau.arr$.

c) Profile: Propose an algorithm for computing all interesting departure times from $src$ to $dst$ where a departure time $\tau.dep$ is interesting if there is a journey reaching $dst$ at some time $\tau.arr$ such that no other journey with departure time $\tau > \tau.dep$ arrives at $\tau.arr$ or before.

We now consider a connected symmetric footpath graph $G$ whose vertex set $V$ contains all stops. We let $d_G(u, v) = d_G(v, u)$ denote the walking time from $u$ to $v$. A traveler can now walk from a stop to any other (unrestricted walking). We still assume that a traveler arriving at stop $u$ at time $\tau$ can catch any connection $c$ such that $c$.from $= u$ and $\tau \leq c.dep$. For representing these walking transfers, we assume that a hub labeling is given: each vertex $u \in V$ is associated to a set $H(u) \subseteq V$ of hubs. Each hub $x \in H(u)$ is associated with its walking time $d_G(u, x)$ from $u$. The hub sets have the following covering property: for any pair $u, v$ of vertices there exists a common hub $x \in H(u) \cap H(v)$ lying
on a shortest walking path from \( u \) to \( v \): 
\[
d_G(u, v) = d_G(u, x) + d_G(x, v) .
\]
We let 
\[
\Delta = \max_{u \in V} |H(u)|
\]
denote the maximum size of a hub set.

a) **Unrestricted walking:** Propose a modification of the algorithm proposed in a) so that journeys can include walking transfer from a stop to any other. How does the complexity of your algorithm increase with \( \Delta \)?

**Solution :**

a) Use Connection Scan without transfers: for each connection \( c \in C \) (in increasing order of departure time), if \( \tau(c.from) \leq c.dep \) then set \( \tau(c.to) := \min(\tau(c.to), c.arr) \). This requires constant time per connection, yielding linear complexity. All the connections of the best trip to \( dst \) must be visited in order and \( \tau(dst) \) is the correct arrival time after all these connections have been scanned.

b) Same algorithm but reversed: Initialize \( \tau(dst) = \tau(arr) \) and \( \tau(u) = -\infty \) for \( u \neq dst \). For each connection \( c \in C \) in reverse order of departure time, if \( c.arr \leq \tau(c.to) \) then set \( \tau(c.dep) := \max(\tau(c.dep), c.dep) \).

c) We proceed similarly as in b) but storing interesting departures at each stop with associated arrival times. For each connection \( c \in C \) in reverse order of departure time, let \( \tau_{dep} \) be the first interesting departure time after \( c.arr \) at \( c.to \) and let \( \tau_{arr} \) be the associated arrival time at \( dst \) (if there is one). Store \( c.dep \) at \( c.from \) with associated arrival time \( \tau_{arr} \) if it is interesting, i.e. no pair \( \tau'_{dep}, \tau'_{arr} \) stored at \( c.from \) satisfies \( \tau'_{dep} > c.dep \) and \( \tau'_{arr} \leq \tau_{arr} \). Dichotomic search can be used if pairs are stored by decreasing departure. The complexity is then within a logarithmic factor from linear.

a') For \( x \in H(src) \), \( \tau(x) := \min(\tau(x), \tau(src) + d_G(src, x)) \).

For each connection \( c \in C \) in increasing order of departure:

- for each \( x \in H(c.from) \), set \( \tau(c.from) := \min(\tau(c.from), \tau(x) + d_G(x, c.from)) \),
- if \( \tau(c.from) \leq c.dep \) then set \( \tau(c.to) := \min(\tau(c.to), c.arr) \),
- for each \( x \in H(c.to) \), set \( \tau(x) := \min(\tau(x), \tau(c.to) + d_G(c.to, x)) \).

For \( x \in H(dst) \), \( \tau(dst) := \min(\tau(dst), \tau(x) + d_G(x, dst)) \).

The covering property ensures correctness. The complexity is linear in \( \Delta \).