Algorithm 6 highest-label($G, s, t$)
1: initialize preflow
2: foreach $u \in V \setminus \{s, t\}$ do
3: \hspace{1em} $u.current-neighbour \leftarrow u.neighbour-list$-head
4: while $\exists$ active node $u$ do
5: \hspace{1em} select active node $u$ with highest label
6: \hspace{2em} discharge($u$)

Lemma 1
When using highest label the number of non-saturating pushes is only $O(n^3)$.

A push from a node on level $\ell$ can only “activate” nodes on levels strictly less than $\ell$.

This means, after a non-saturating push from $u$ a relabel is required to make $u$ active again.

Hence, after $n$ non-saturating pushes without an intermediate relabel there are no active nodes left.

Therefore, the number of non-saturating pushes is at most $n(\#\text{relabels} + 1) = O(n^3)$.

Since a discharge-operation is terminated by a non-saturating push this gives an upper bound of $O(n^3)$ on the number of discharge-operations.

The cost for relabels and saturating pushes can be estimated in exactly the same way as in the case of the generic push-relabel algorithm.

Question:
How do we find the next node for a discharge operation?

Maintain lists $L_i$, $i \in \{0, \ldots, 2n\}$, where list $L_i$ contains active nodes with label $i$ (maintaining these lists induces only constant additional cost for every push-operation and for every relabel-operation).

After a discharge operation terminated for a node $u$ with label $k$, traverse the lists $L_k, L_{k-1}, \ldots, L_0$, (in that order) until you find a non-empty list.

Unless the last (non-saturating) push was to $s$ or $t$ the list $k - 1$ must be non-empty (i.e., the search takes constant time).
13.3 Highest Label

Hence, the total time required for searching for active nodes is at most

\[ O(n^3) + n(\#\text{non-saturating pushes to } s \text{ or } t) \]

**Lemma 2**

The number of non-saturating pushes to \( s \) or \( t \) is at most \( O(n^2) \).

With this lemma we get

**Theorem 3**

The push-relabel algorithm with the rule highest-label takes time \( O(n^3) \).

---

**Proof of the Lemma.**

- We only show that the number of pushes to the source is at most \( O(n^2) \). A similar argument holds for the target.
- After a node \( v \) (which must have \( \ell(v) = n + 1 \)) made a non-saturating push to the source there needs to be another node whose label is increased from \( \leq n + 1 \) to \( n + 2 \) before \( v \) can become active again.
- This happens for every push that \( v \) makes to the source. Since, every node can pass the threshold \( n + 2 \) at most once, \( v \) can make at most \( n \) pushes to the source.
- As this holds for every node the total number of pushes to the source is at most \( O(n^2) \).