Supplementary material

Maintaining sliding-window neighborhood profiles in interaction networks

Rohit Kumar[1], Toon Calders[1], Aristides Gionis[2], and Nikolaj Tatti[2]

[1]Department of Computer and Decision Engineering
Université Libre de Bruxelles, Belgium

[2]Helsinki Institute for Information Technology
Aalto University, Finland

A Correctness and complexity of the exact algorithm

Proposition 1. AddEdge *updates the summary correctly.*

Proof. Assume that we are adding $\{a, b\}$ at time t, and let H be the shapshot graph before adding this edge. Fix x. Let us define $\alpha_v(i) = h_H(x, v, i)$. Similarly, define $\beta_v(i) = h_{(t+1)}(x, v, i)$. To prove the proposition we need to show that *(1)* $\beta_v(i) = \max(g(v, x, i), \alpha_v(i))$ and *(2)* if $g(v, x, i)$ is not set, then $\alpha_v(i) = \beta_v(i)$.

Let us first prove that whenever set, we maintain the invariant,

$$g(v, x, i) \leq \max\left\{h(p) \mid p \in Q_v, |p| - 1 \leq i\right\} \leq b_v(i), \tag{1}$$

where Q_v contains all paths from x to v in $G(t + 1)$ containing (a, b) or (b, a). Note that the second inequality follows immediately from the definition of β. We prove the first by induction over i. The case $i = 1$ is trivial. If $i > 1$, then if $g(v, x, i)$ is set, then either it is set by AddEdge or there is w such that $g(w, x, i - 1)$ is set. In the first case and, due to induction assumption, in the second case, it follows that $g(v, x, i)$ is a horizon of some path in Q_v of length at most i.

We prove the main claim also by induction over i. Assume $i = 1$. The initialization of $g(v, x, 1)$ in AddEdge now guarantees *(1)* and *(2)*.

Assume $i > 1$. Assume that $\beta_v(i) > \alpha_v(i)$. This can only happen if there is a path $p = \langle v_0, \ldots, v_k\rangle \in Q_v$ with $h(p) = \beta(i)$. Let $p' = \langle v_0, \ldots, v_{k-1}\rangle$ and let $w = v_{k-1}$. We must have $\alpha_w(i - 1) < \beta_w(i - 1)$, as otherwise we have $\beta_v(i) = \alpha_v(i)$. By induction, *(1)* immediately implies that $\beta_w(i-1) = g(w, x, i-1) > \alpha_w(i-1)$. This means that Merge$(x, w, g(w, x, i - 1), i - 1)$ is called, and it returns true. Consequently, $g(v, x, i) \geq h(p) = \beta_v(i)$, Eq. 1 implies that $g(v, x, i) = \beta_v(i)$. This immediately proves *(1)* and *(2)*. □

Proposition 2. *Let* $n = |V|$, $m = |E|$, *and* r *be the upper bound on the distances we are maintaining. The time complexity of* AddEdge *is* $\mathcal{O}(rmn \log(n))$. *The space complexity is* $\mathcal{O}(rn^2)$.

Proof. The complexity of Algorithm 3 is $\log(n)$, since we need to search a summary and update $S^v[i]$ for node x.

Every $g(u, x, i+1)$ will be initiated only if $g(v, x, i)$ was set for one of its neighbors v. As such, this may happen at most as many times as u has neighbors in the graph. Since the cummulative sum of all neighbors is $2m$ we can hence bound the number of times a $g(u, x, i+1)$ is set for x to $2m$. Since there are n nodes, lines 5,6,7 are executed at most $2nm$ times per length i, and as a consequence this is also an upper bound on the number of calls to Algorithm 3. Putting it all together, we get a complexity of $\mathcal{O}(2nmr(\log(n)))$ for Algorithm 2. Since Algorithm 1 does only call Algorithm 2 once, this proves the complexity bound for time.

The complexity bound on space easily follows from the observation that for every node v, and every distance $i = 0, \ldots, r$, the summary $S^v[i]$ contains at most one entry for any other node. $\square$

B Correctness and complexity of the sketch algorithm

Proposition 4 (Part 1). *The sketch version of* AddEdge *updates the summary correctly.*

Proof. Assume that we are adding $\{a, b\}$ at time t, and let H be the shapshot graph before adding this edge. Fix x. Let us define $\alpha_v(i) = t$, where $(x, t) \in C^v[i]$ (based on H), and ∞ otherwise. Similarly, define $\beta_v(i)$ using $G(t+1)$. To prove the proposition we need to show that *(1)* $\beta_v(i) = \max(g(v, x, i), \alpha_v(i))$ and *(2)* if $g(v, x, i)$ is not set, then $\alpha_v(i) = \beta_v(i)$.

Let us first prove that whenever set, we maintain the invariant

$$g(v, x, i) \leq \max\{h(p) \mid p \in Q_v, |p| - 1 \leq i\} \leq b_v(i), \tag{2}$$

where Q_v contains all paths from a node y, with $\rho(y) = x$, to v in $G(t+1)$ containing (a, b) or (b, a). The second inequality follows immediately by definition of β. We prove the first by induction over i. The case $i = 1$ is trivial. If $i > 1$, then if $g(v, x, i)$ is set, then either it is set by AddEdge or there is w such that $g(w, x, i-1)$ is set. In the first case and, due to induction assumption, in the second case, it follows that $g(v, x, i)$ is a horizon of some path in Q_v of length at most i.

We prove the main claim also by induction over i. Assume $i = 1$. The initialization of $g(v, x, 1)$ in AddEdge now guarantees *(1)* and *(2)*.

Assume $i > 1$. Assume that $\beta_v(i) > \alpha_v(i)$. This can only happen if there is a path $p = \langle v_0, \ldots, v_k \rangle \in Q_v$ with $h(p) = \beta(i)$. Let $p' = \langle v_0, \ldots, v_{k-1} \rangle$ and let $w = v_{k-1}$. We must have $\alpha_w(i-1) < \beta_w(i-1)$, as otherwise we have $\beta_v(i) = \alpha_v(i)$. By induction, *(1)* immediately implies that $\beta_w(i-1) = g(w, x, i-1) > \alpha_w(i-1)$. This means that Merge$(x, w, g(w, x, i-1), i-1)$ is called, and it returns true. Consequently, $g(v, x, i) \geq h(p) = \beta_v(i)$, Eq. 2 implies that $g(v, x, i) = \beta_v(i)$. This immediately proves *(1)* and *(2)*. $\square$

The next proposition gives an upper bound on space and memory consumption of the algorithm.

Proposition 4 (Part 2). *Let $n = |V|$, $m = |E|$, and r be the upper bound on the distances we are maintaining. The time complexity of the sketch version of* AddEdge *is $\mathcal{O}(2^k rm \log^2(n))$. The space complexity is $\mathcal{O}(2^k nr \log^2 n)$.*

Proof. Algorithm 4 needs to visit the iterate the entries in $C^v[i]$. Since there are at most $\mathcal{O}(\log n)$ different values of ρ, there are at most $\mathcal{O}(\log n)$ entries.

Every $g(u, x, i + 1)$ will be initiated only if $g(v, x, i)$ was set for one of its neighbors v. As such, this may happen at most as many times as u has neighbors in the graph. Since the cummulative sum of all neighbors is $2m$ we can hence bound the number of times a $g(u, x, i + 1)$ is set for x to $2m$. Since there are $\mathcal{O}(\log n)$ different values of ρ, lines 5,6,7 are executed at most $\mathcal{O}(\log nm)$ times per length i, and as a consequence this is also an upper bound on the number of calls to Algorithm 3. Putting it all together, we get a complexity of $\mathcal{O}(2mr \log^2(n))$ for Algorithm 2. Since Algorithm 1 does only call Algorithm 2 once, this proves the complexity bound for time.

The complexity bound on space easily follows from the observation that for every node v, and every distance $i = 0, \ldots, r$, the summary $C^v[i]$ contains at most $\mathcal{O}(\log n)$ entries that, and each entry requires $\mathcal{O}(\log n)$ space. □

C Code to run the experiments

Please download the instruction and code to run the experiments from the below link:

Download Experiment Code