Sparsity – homework 3

Michał Zawalski

November 22, 2019

1 Problem 1.

Let $c = wcol_p(\mathcal{C})$. Since $\mathcal{C}$ has bounded expansion, this constant is well-defined. Take any $G \in \mathcal{C}$. Let σ be any $wcol_p$-optimal ordering of G. Now let's consider D to be the graph such that $V(D) = V(G)$ and for every vertex $v \in V(D)$ we have $(v, a) \in E(D) \Leftrightarrow a \in WReach_p[G, \sigma, v]$. Clearly outdegree is bounded by c. Therefore it remains to show that every connected subgraph of size at most p points to one vertex. Take any connected subgraph $A \subseteq G$ such that $|A| \leq p$. Let $a \in A$ be the σ-minimal vertex of A. Clearly every vertex of A has a G-path to a consisting of vertices from A and thus of length at most p. Since a is σ-minimal, this path witnesses that a belongs to $WReach$ of every vertex in A, and thus there is an edge from every vertex of A to a, which completes the proof.

2 Problem 2.

As proven on lectures, in polynomial time we can compute ordering σ of G such that $adm_d(G, \sigma) \leq d \cdot adm_d(G)$. Then

$$wcol_d(G, \sigma) \leq 1 + d(adm_d(G, \sigma) - 1)^{d^2} \leq 1 + d(d \cdot adm_d(G) - 1)^{d^2}$$

and since $\mathcal{C}$ has bounded expansion, it has bounded d-admissibility, which means that according to the given inequality gives $wcol_d(G, \sigma) \leq c$, where c is a constant depending only on d and $\mathcal{C}$.

Now I will show the algorithm for *Encoder*: firstly, given G he computes the ordering σ of G mentioned above. Then he labels the consecutive vertices in this ordering with natural numbers $1, 2, \ldots, n$. For every vertex v he computes $WReach_d[G, \sigma, v]$ and outputs a function $\lambda(v) := \{(u, [distance\ from\ v\ to\ u]) \mid u \in WReach_d[G, \sigma, v]\}$ (concatenation of entries in this set). Clearly this algorithm works in Ptime, since, as mentioned before, σ can be computed in Ptime, relabelling and computing $WReach$ can also clearly be done in Ptime. Moreover, the labels have length at most $2c \cdot \log n = O(\log n)$.

Given $\lambda(v), \lambda(u)$, the *Decoder* do as follows: for every element $((x, d_1), (y, d_2)) \in \lambda(v) \times \lambda(u)$, if $x = y$ it computes $d_1 + d_2$ and chooses the minimal value obtained this way. If no such pair was found or minimal score is greater than d, it outputs ∞. Clearly it works in Ptime with respect to input length. It remains to show the answer is correct. Let P be the shortest path between u and v. Let $a \in P$ be its σ-minimal vertex. If $|P| \leq d$, then $dist_G(u, a), dist_G(v, a) \leq d$. Clearly there are shortest paths both between u-a and v-a that consists only of vertices from P, and by minimality of a these paths witness that a belongs to $WReach_d$ of both vertices. Therefore it will be found.

Note that for any vertex $g \in G$ we have $dist_G(u,a) + dist_G(v,a) \leq dist_G(u,g) + dist_G(v,g)$ and thus this value will be the smallest considered, which gives the correct answer. If $|P| > d$, then similarly, for any $g \in G$ we have $dist_G(u,g) + dist_G(v,g) > d$ and thus the algorithm will output ∞. Therefore the answer will always be correct, which completes the proof.

3 Problem 3.

Firstly, recall the Sunflower Lemma:

Lemma 1. *Let $\mathcal{F}$ be a family of sets from universe U with cardinality exactly n, without duplicates. If $|\mathcal{F}| > n!k^n$, then $\mathcal{F}$ contains a $(k+1)$-petal **sunflower**: a subfamily $F \subseteq \mathcal{F}$ of size $k+1$ such that $\forall_{1 \leq i < j \leq k+1} F_i \cap F_j = \bigcap_{i \in [1,k+1]} F_i$. Moreover, it can be found in polynomial time with respect to size of $\mathcal{F}, U$ and k.*

Proof of this lemma can be found in [1].

If $m = 0$, the problem is trivial. Hence I will assume that $m \geq 1$. Consider a function $F : A \to \mathcal{P}(V(G))$ defined in the following way: if $|WReach_r[G,\sigma,v]| = c$, let $F(v) = WReach_r[G,\sigma,v]$; otherwise obtain $F(v)$ by taking $WReach_r[G,\sigma,v]$ and adding there $c - |WReach_r[G,\sigma,v]|$ additional fake elements. Observe that if $a <_\sigma b$, then $b \in F(b), b \notin F(a)$ and thus $F(a) \neq F(b)$. This way we have $|A|$ sets of cardinality exactly c in the universe of size at most $(c+1)|V(G)|$, which are all polynomial in the size of input. From the assumption $|A| \geq 4(2cm)^{c+1} = 4c^c \cdot c(2m)^{c+1} > c!m^c$. Thus, according to the Sunflower Lemma, there is a $(m+1)$-petal sunflower $\mathcal{X}$, denote its sets by X_v in such a way that $X_v = F(v)$. Let $S = \bigcap X_v$ and let $B = \{a \mid a \in A \wedge F(a) \in \mathcal{X}\}$ without the σ-minimal element. I will show that S, B defined this way gives a solution for the problem. Clearly they can be computed in Ptime, since according to the Sunflower Lemma we can compute $\mathcal{X}$ in Ptime.

Since all the sets X_v are of size c, then $|S| \leq c$. Note that S consists of vertices of G only, because no fake element was used twice. Since $\mathcal{X}$ is $(m+1)$-petal sunflower, $|B| = m$. Clearly $B \subseteq Dom(F) = A$. Now I will show that B is distance-r independent. Suppose by way of contradiction that there are $a, b \in B$ such that $|P_{ab}| \leq r$, where $P_{ab} \subseteq G - S$. Let v be the σ-minimal element of P_{ab}. Let P_{av}, P_{bv} be the parts of P_{ab} after splitting in v. By σ-minimality of v, these paths witness that v belongs both to $WReach_r[G,\sigma,a]$ and $WReach_r[G,\sigma,b]$. Therefore it also belongs to the core of sunflower, which means that $v \in S$, which gives contradiction. Finally, it remains to show that $B \cap S = \emptyset$. Let a be any element of B. Let b be the σ-minimal element of $\{a \mid a \in A \wedge F(a) \in \mathcal{X}\}$. By the construction of B, we know that $a \neq b$ and $b <_\sigma a$. Therefore clearly $a \notin WReach[G,\sigma,b]$ and thus a cannot be in the sunflower's core, which is equal to S. This completes the proof.

References

[1] Marek Cygan, Fedor V. Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, Saket Saurabh *Parameterized algorithms, http://parameterized-algorithms.mimuw.edu.pl/parameterized-algorithms.pdf.*