# London Taught Course Centre

**2019 examination**

# Graph Theory

## Answers

**1** There was a small (not deliberate) error in the exam paper; the condition in 'labelled topological minor' should have been $\phi(v) \prec v$, not the other way round; if taken literally this would make the question wrong. But most people read what 'should' be there and gave a more or less correct proof of what i intended.

This is a standard theorem (Google will lead you to it directly); it's more or less what Kruskal actually proved (the version in the lecture notes, week 2, of unlabelled trees, is equivalent to the $|S| = 1$ case).

The proof is an easy modification of the lecture notes proof; one strengthens the statement to ordered labelled topological minor on rooted labelled trees (as in the lecture notes) and follows the proof there, adding 'labelled' wherever it is necessary. The only point where one needs to do more is that it does not suffice to find one $(i, j)$ such that the rooted forest $T_i - r_i$ is an ordered labelled topological minor of $T_j - r_j$, since we cannot simply map $r_i$ to $r_j$; the label of $r_i$ need not precede the label of $r_j$. But by Propositions 6 and 8 of the lecture notes, at this point in the proof it follows that there is actually some infinite chain $i_1 < i_2 < \dots$ such that $T_{i_a} - r_{i_a}$ is an ordered labelled topological minor of $T_{i_b} - r_{i_b}$ whenever $a < b$. Since the labels are well-quasi-ordered, the sequence of labels of $r_{i_1}, r_{i_2}, \dots$ cannot be an infinite strictly descending sequence or infinite antichain; so there exist $a < b$ such that $r_{i_a}$ has label preceding that of $r_{i_b}$, and we thus have the desired fact that $T_{i_a}$ is an ordered labelled topological minor of $T_{i_b}$.

**2** This is also quite standard, though one has to guess a little what to type to Google. Warning: you can easily miss research which is very relevant but uses terminology which you don't expect or which is different to what you use in your field. 'Tangled' is usually called 'intrinsically knotted'.

(a) Suppose $G$ is not tangled; then there is a PL-embedding $\phi$ which witnesses this. Now when we delete edges or vertices from $G$ to form a new graph $H$, trivially the restriction of $\phi$ to $H$ has the property that all cycles in $H$ are unknotted: a cycle in $H$ is also a cycle in $G$ and $\phi$ witnesses that $G$ is not tangled. If we contract an edge to obtain $H$, it is also fairly trivial to modify $\phi$ to witness that $H$ is not tangled; we extend the edges going to $u$ to follow $\phi_{uv}$ to $v$, without either creating new intersections (which would stop the result being an embedding) or a knotted cycle. I'd be happy with this assertion, but it's actually not completely trivial to give a construction which works. If you simply 'follow $\phi_u v$ to $v$' linearly, you may cause two edges to intersect which should not (if $\phi$ is not a generic embedding witnessing untangledness) but if you make a too arbitrary choice you risk creating a new knot. One possibility which almost surely works is to start by perturbing $u$ (add a uniform random vector from the $\varepsilon$-ball about the origin, where $\varepsilon > 0$ is sufficiently small) and appropriately alter the final linear segments of edges going to $u$), then terminate these edges on the ball $B$ of radius $\varepsilon$ about $u$, and follow the termination points as $B$ is moved along $\phi_{uv}$ and scaled linearly to zero radius. Justifying this is easy but tedious.

What this shows is that the class of graphs which are not tangled is closed under taking minors; by the Robertson-Seymour Theorem from notes, there is thus a finite collection of forbidden minors $F_1, \dots, F_t$. And it is obviously possible to check if a given $G$ contains a given $F_i$ as a minor or not in finite time by brute force (in fact, there is even a polynomial time algorithm, so the decision question is actually not just decidable but also in P).

(b) There is no such Turing Machine. There exist graphs which are tangled (such as $K_7$); a Google search will find this. And there exist graphs which are not tangled (such as the empty graph). Since a Turing Machine can only look at the first 2019 entries of the adjacency matrix of any graph, it's enough to take $G$ to be the union of $K_7$ and enough isolated vertices that the first 2019 entries of its adjacency matrix are all zero; then $G$ cannot be distinguished from the empty graph on $v(G)$ vertices in time 2019, but one is tangled and the other is not.

## 3

(a) Summing over $v$, we have $\sum_{v\in V(H)} \sum_{\substack{T\in K_3(H)\\ v\in T}} w(T) \leq v(H)$; since each $K_3$-copy appears exactly three times in the sum on the left, we have $3\sum_{T\in K_3(H)} w(T) \leq v(H)$ and the desired inequality follows.

(b) Let $R(G)$ be as in the question. Given a fractional triangle factor $w$ of $R(G)$ with weight at least 0.333, we begin the following algorithm. Start with $y$ the everywhere zero fractional triangle factor of $R(G)$ and $S$ the empty set. Choose $T \in K_3\big(R(G)\big)$ such that $y(T) \leq 0.9999w(T)$, and pick a triangle in $G$ with one vertex in each part indexed by $T$ which is vertex-disjoint from the triangles in $S$. Add this triangle to $S$, and increase $y(T)$ by $\frac{1}{|V_1|}$. Repeat until $y(T) \geq 0.9999w(T)$ for every $T \in K_3\big(R(G)\big)$.

This algorithm can only fail if at some point it is not possible to pick a triangle in $G$ as stated. But by definition of a triangle factor, and since $n$ is large enough, at every step the number of vertices in each part of $G$ which are not used in $S$ is at least $10^{-6}\frac{n}{k}$, and by definition of a regular pair, for any $T \in K_3\big(R(G)\big)$, the three sets $\big(V_i \setminus \bigcup S\big)_{i\in T}$ form $10^7\varepsilon$-regular pairs of density at least $d/2$. By a theorem from the course they therefore contain a triangle provided $\varepsilon$ is small enough.

Thus the algorithm completes, and by definition of a fractional triangle factor and construction we see $S$ is a set of at least $0.33n$ pairwise vertex-disjoint copies of $K_3$ in $G$.