ETH Zürich
Institute of Theoretical Computer Science
Dr. J. Lengler
H. Einarsson, R. Nenadov

FS 2013
Proposed solutions for sheet 2

Graphs and Algorithms

Solutions

Exercise 1 (Large bipartite subgraph)

We will create the partitioning by adding the vertices of V to A or B one by one. Let us assume V' is the set of vertices we have already added and at least half of the edges in $G[V']$ are in between A and B. Now if u is the next vertex to insert we will *greedily* add it to A if $|\Gamma(u) \cap B| > |\Gamma(u) \cap A|$ otherwise we add it to B. Thus after inserting u the invariant that at least half of the edges in $G[V' \cup \{u\}]$ still holds so adding the vertices greedily using this method will result in at least half of the edges being between A and B when we have inserted all of V.

Exercise 2 (Graph isomorphisms)

(a) $\theta\alpha$ is bijective since it is the composition of two bijective maps. We have

$$\{u,v\} \in E(G) \overset{\text{(i)}}{\iff} \{\alpha(u),\alpha(v)\} \in E(G) \overset{\text{(ii)}}{\iff} \{\theta(\alpha(u)),\theta(\alpha(v))\} \in E(H)$$

because (i) α is an automorphism, and (ii) θ is an isomorphism. Taking both equivalences together, $\theta\alpha$ is an isomorphism.

(b) Let $\mathcal{I}$ be the set of all isomorphisms from G to another graph. (a) proves that $\theta\mathrm{Aut}(G) \subseteq \mathcal{I}$.

To show $\mathcal{I} \subseteq \theta\mathrm{Aut}(G)$, take any $\phi \in \mathcal{I}$. Then $\theta^{-1}\phi$ is an automorphism on G: clearly it exists and is bijective. Similar to (a) we have

$$\{u,v\} \in E(G) \iff \{\phi(u),\phi(v)\} \in E(H) \iff \{\theta^{-1}\phi(u),\theta^{-1}\phi(v)\} \in E(G).$$

But this means $\theta^{-1}\phi \in \mathrm{Aut}(G)$, hence $\phi \in \theta\mathrm{Aut}(G)$.

(c) Let $\mathcal{P}$ be the set of permutations of $V(G)$. Thus $|\mathcal{P}| = n!$. Let $\sim$ be the equivalence relation given by $\pi \sim \tau$ iff $\pi^{-1}\tau$ is an automorphism on G. Then the equivalence class $[\mathrm{id}] \in \mathbb{P}/\sim$ of the identity permutation id is exactly $\mathrm{Aut}(G)$, and because of (b) each equivalence class is the set of isomorphisms from G to some H, and in particular all classes are of order $|\mathrm{Aut}(G)|$. Thus we have

$$n! = |\mathrm{Aut}(G)| \cdot |G/\sim| = |\mathrm{Aut}(G)| \cdot \{H \text{ isomorphic to } G\}$$

as desired.

(d) Let us define $\mathcal{G}_n$ as the set of all isomorphism classes of graphs on n vertices and let us define $h(n)$ as the number of labelled graphs on n vertices. We know that $h(n)$ is equal to $2^{\binom{n}{2}}$ since for each edge we either include it or not. Now

$$\begin{aligned} g(n) &= \sum_{g \in \mathcal{G}_n} 1 \\ &= n! \sum_{g \in \mathcal{G}_n} \frac{1}{n!} \\ &\leq n! \sum_{g \in \mathcal{G}_n} \frac{1}{\#\mathrm{Aut}(g)} \\ &= h(n) \end{aligned}$$

which gives the upper bound and for the lower bound we have that

$$\begin{aligned} n! \cdot g(n) &= n! \cdot \sum_{g \in \mathcal{G}_n} 1 \\ &\geq n! \sum_{g \in \mathcal{G}_n} \frac{1}{\#\mathrm{Aut}(g)} \\ &= h(n). \end{aligned}$$

We now have that

$$\log\left(2^{\binom{n}{2}}\right) = \binom{n}{2} \cdot \log(2) \quad \text{and} \quad \log(n!) = \sum_{i=1}^{n} \log(i) \leq n \log(n).$$

That is

$$\binom{n}{2} \cdot \log(2) - n \log(n) \leq \log(g(n)) \leq \binom{n}{2} \cdot \log(2)$$

which yields $\log(g(n)) = \frac{n^2}{2} \cdot \log(2) + O(n \cdot log(n))$ which is the desired result.

Remark: Actually, almost all graphs have trivial automorphism group, so that $g(n)$ is combinatorially equivalent to $\frac{1}{n!} 2^{\binom{n}{2}}$. The proof requires combinatorial means that are beyond the scope of this course.

Exercise 3 (Tree Counting)

(a) The number of labelled trees with a prescribed degree sequence $d_1, \ldots, d_n$ is

$$\frac{(n-2)!}{\prod_{i=1}^{n}(d_i - 1)!} \ .$$

This can be seen as follows. First observe that each vertex i appears $d_T(i) - 1$ times in the Prüfer code of a tree T: A non-leaf vertex v appears one time for each deleted neighbor. When v has only one remaining neighbor, it is either deleted or remains with the last edge. Leaves are not recorded at all.

To count trees with prescribed degree sequence $d_1, \ldots, d_n$ we can therefore count lists of length $n-2$ that contain $d_i - 1$ entries of vertex i for each $i = 1, \ldots, n$. For every vertex i we make its occurences in the list distinguishable by introducing subscripts to each occurence. Then we are permuting $n-2$ distinct objects, namely $\{1_1, \ldots, 1_{d_1-1}, \ldots, n_1, \ldots, n_{d_n-1}\}$, and there are $(n-2)!$ lists with these objects. Since every entry i_j in the list refers to the same vertex i, we counted each desired arrangement $\prod_{i=1}^{n}(d_i-1)!$ times, once for each permutation of the subscripts on each vertex.

(b) There are

$$\binom{n}{3} \cdot (n-3) \cdot \frac{(n-2)!}{2}$$

labelled trees with three leaves. This can be seen as follows. Each such tree corresponds to a Prüfer code in which exactly three vertices do not appear, and hence exactly one vertex appears twice. This shows that in each such tree there are three vertices of degree 1, one vertex of degree 3, and all remaining vertices have degree 2. There are $\binom{n}{3}\cdot\binom{n-3}{1} = \binom{n}{3}\cdot(n-3)$ degree sequences with this property. And by (a) each such sequence gives us $\frac{(n-2)!}{2}$ trees.

(c) Obviously, the number of trees on n vertices that contain the edge e is the same for every fixed edge $e \in \binom{[n]}{2}$. (The choice of the edge $\{3,4\}$ has no influence on the result. Asking for the number of trees containing the edge $\{1,4\}$ yields the same number.) Combining that

- there are in total n^{n-2} trees on n vertices by Cayley's formula,
- there are in total $\binom{n}{2}$ edges on n vertices, and
- every tree has $n-1$ edges

we obtain that the number of trees containing any fixed edge e is

$$\frac{n-1}{\binom{n}{2}} \cdot n^{n-2} = \frac{2}{n} \cdot n^{n-2} = 2n^{n-3}.$$

(We tacitly imply that $n \geq 4$. Otherwise there are clearly no trees containing the edge $\{3,4\}$.)

(d) Let $K_{n,m}$ be a labeled complete bipartite graph with partition sets A and B of size n and m respectively. We say that a word $w' \in S^{k'}$ is a subword of $w \in S^k$ (or, equivalently, w contains w') if there exists an injective function $f_I : [k'] \to [k]$ such that $w'_i = w_{f_I(i)}$, for all $i \in [k']$, and further $f_I(i) < f_I(i+1)$ for all $i \in [k'-1]$. We prove the following,

1. in the Prüfer code of a spanning tree of $K_{n,m}$, there are exactly $n-1$ elements from B and $m-1$ elements from A,
2. for any $\mathbf{a} \in A^{m-1}$ and $\mathbf{b} \in B^{n-1}$, there exists a unique Prüfer code $w \in [n+m]^{n+m-2}$ which corresponds to a spanning tree of $K_{n,m}$ and contains $\mathbf{a}$ and $\mathbf{b}$.

Note that these two observations yield that there are

$$n^{m-1} \cdot m^{n-1}$$

spanning trees of a labelled $K_{n,m}$.

We first prove the claim (1). This easily follows from an inspection of the algorithm: each time when we remove a vertex from the given tree, we append the label of its neighbour (in the remaining tree) to the output word. Thus, whenever we remove a vertex from A we append some vertex $b \in B$, and vice versa. On the other hand, since each edge goes between A and B, and we know that there exists an edge between the last two remaining vertices, it follows that it is not possible that both of those vertices belong either to A or B. Therefore, during the algorithm we have removed all but one vertex from A and all but one vertex from B, thus the claim follows.

Next we prove the claim (2). We will, by induction on the number of iterations of the inverse algorithm, construct the word $w \in [n+m]^{n+m-2}$. In the first step of the algorithm we have $S = \varnothing$, and let $s_1 \in [n+m]$ be the smallest number that doesn't appear in either **a** or **b**. Note that we trivially have $w_1 \in \{a_1, b_1\}$, as otherwise w coulnd't contain both **a** and **b**. If $s_1 \in A$ and $w_1 = a_1$, then the Prüfer code corresponds to a tree which contains an edge $\{s_1, a_1\}$, which cannot be since $K_{n,m}[A]$ is an independent set. Therefore, if $s_1 \in A$ then $w_1 = b_1$, and similarly $w_1 = a_1$ if $s_1 \in B$, thus w_1 is uniquely determined. Let us now assume that we have uniquely determined the first $t - 1 \leq m + n - 3$ elements of w, such that it contains the first t_a elements of **a** and t_b elements of **b**. Let s_t be the smallest element of $[n+m] \setminus S$. By the same argument we have $w_t \in \{a_{t_a+1}, b_{t_b+1}\}$, and further $w_t = a_{t_a+1}$ if $s_t \in B$ and $w_t = b_{t_b+1}$ if $s_t \in A$. Therefore, there exists a unique Prüfer code $w \in [n+m]^{n+m-2}$ that corresponds to a spanning tree of $K_{n,m}$ and contains both **a** and **b**.

Exercise 4 (Block graphs are trees)

Let $G = (V, E)$ be a connected graph and let $\mathcal{B}(G) = (A \cup B, E')$ denote its block graph, where $A \subset V$ is the set of articulation points, B the set of blocks and $E' = \big\{\{a, b\} \mid a \in A, b \in B, a \in V(b)\big\}$.

As said in the lecture, in order to prove the exercise we may use the following fact without a proof.

Fact 1. *Let $\approx \subseteq E \times E$ be a relation such that $e_1 \approx e_2$, $e_1, e_2 \in E$, if and only if there exists a cycle in G which contains both e_1 and e_2. Then $\approx$ is an equivalence relation.*

Proof. In order to make the proof of the exercise complete, we also give a proof of this fact. Clearly $\approx$ is symmetric and reflexive, thus the only thing left to show is that it is transitive. Let $e_1 = \{v_1, v_2\}$, $e_2 = \{u_1, u_2\}$ and $e_3 = \{w_1, w_2\}$ be such that $e_1 \approx e_2$ and $e_2 \approx e_3$. Since $e_1 \approx e_2$, there exist at least two disjoint paths between sets e_1 and e_2 (we interchangebly use that edge is a set of two elements), and thus by Menger's theorem the minimum size of a $e_1 - e_2$ separating set is at least 2. Similarly, we get that a $e_2 - e_3$ separating set is also

of size at least 2. Therefore, by removing any vertex $v \in V$ we get, without loss of generality, that there exists a path from v_1 to u_1 and also from u_1 to w_1. In particular, this imples that there exists a walk from v_1 to w_1, and thus a path, which avoids v. Hence $\{v\}$ is not $e_1 - e_3$ separating set, and it holds for every $v \in V$. Therefore, again by Menger's theorem we have that there are at least two disjoint $e_1 - e_3$ paths, thus there exists a cycle which contains both e_1 and e_3. □

Proof of (1). The statement is true for blocks which do not intersect. Assume now that two blocks b_1, b_2 intersect in at least two vertices v and w. If b_1 (b_2, respectively) contains at least two edges, then there exists a path between v and w which uses only edges in b_1 and avoids edge $\{v, w\}$ (if such exists). To see this, consider edges $e_v, e_w \in b_1$ incident to v and w. Then, following the definition, there exists a cycle in b_1 which contains e_v and e_w, and thus v and w. This cycle induces two paths, and at least one of these paths doesn't contain edge $\{v, w\}$. We can assume that either b_1 or b_2 contains at least two edges. Otherwise, since v and w belong to a subgraph induced by edges from b_1 and b_1, and $|b_1| = |b_2| = 1$, we get $b_1 = b_2$. Further, without loss of generality we may assume that b_1 contains at least two edges. On the other hand, we have a path from v to w which uses only edges from b_2 (there exists either a direct edge or b_2 contains at least two edges). However, this path together with a path from v to w which uses only edges from b_1 forms a cycle, again contradiction the maximality of blocks.

It remains to show that if the intersection consists of exactly one vertex v, then this vertex is an articulation point. Let w_1 and w_2 be vertices of b_1, b_2 adjacent to v. If v is not an articulation point, then $G \setminus v$ is connected and contains a path from w_1 to w_2 not using v. We can turn this path into a cycle adding the edges $\{w_1, v\}$ and $\{v, w_2\}$. However this cycle uses edges of both b_1 and b_2, again a contradiction to their maximality. □

Proof of (2). Assume that at least one edge is contained in no block. But then by definition it is a block itself. If one edge is contained in two blocks then their intersection contains at least two vertices, which contradicts part (1). □

Proof of (3). Assume that this is not the case. Then there are two vertices x, y of $\mathcal{B}(G)$ which are not in the same component. Take any vertex $v \in x$ and $w \in y$ in G. As G is connected there is a path P from v to w. This path consists of edges and by part (2) we can map each edge uniquely to a block. Therefore we can segment P in smaller paths which are completely contained in one block. The intersection between subsequent path-segments must be an articulation point by part (1). But this means that we can map P to a path P' in $\mathcal{B}(G)$ by simply taking the alternating $B - A$ path that starts with the block corresponding to the first path segment, goes to the intersection of the first and second path segment, then the block corresponding to the second segment and so on. Note that P' must not yet necessarily start on x and end on y. This is the case if v or w are articulation points. But then we can trivially extend P' to include x and y by either adding them directly to the path (if they are articulation points) or by adding the articulation point between v and the end of the path (i.e. x) and v itself (and equivalently for w). □

Proof of (4). Assume that $\mathcal{B}(G)$ contains a cycle C. As $\mathcal{B}(G)$ is bipartite this cycle must have length at least 4 and contain at least two blocks b and b_1. Remove b from C and denote with $P' = (a_0, b_1, \ldots, a_n)$ the resulting path. Each 3 consecutive vertices of P' of the form (a_1, b, a_2) where $a_1, a_2 \in A$ and $b \in B$ can be mapped to a path from a_1 to a_2 in G only using edges of b. By concatenating all these paths we obtain a path P in G from a_0 to a_n which does not use any edges of b. Note that P' must contain at least one block $b_1 \neq b$ and therefore P must contain some edges which are not part of b. But then we have two edges e_1, e_2, such that $e_1 \in b$ and $e_2 \notin b$ and $e_1 \approx e_2$, which contradicts the maximality of blocks. □

Exercise 5 (Cops and robber 2)

We first show that a single cop cannot catch a robber for $k = 2, 3$ and $n \geq 2$ by giving a winning strategy of the robber. Color the grid in a black and white as you would do for a chessboard (formally, a position is white if the sum of its coordinate is even, otherwise it is odd). Then no two positions of the same color are adjacent to each other. The cop starts by picking some position. Then the robber picks a different field of the same color. Such a field exists whenever $n \geq 2$. Whenever the cop moves, then the robber moves to an arbitrary neighbor which is not occupied by the cop. (This is possible since each field has at least two neighbors). Whenever the cop stands still, the robber does so as well. In this way, after the robber's move his position will have the same color as the position of the cop, so in particular they are not adjacent. Thus the cop cannot catch the robber.

Now we prove that two cops suffice to catch the robber. Let $c = (c_1, c_2, c_3)$ and $d = (d_1, d_2, d_3)$ be the position of the cops, and $r = (r_1, r_2, r_3)$ be the position of the robber. (So they are changing over time.) We will call the first, second, and third entry the x-,y-, and z-dimension, respectively. Note that we include the case $k = 2$ by setting the third coordinate to 1. Thus the proof will work for both $k = 3$ and $k = 2$ at the same time.

We will start with cops at $c = (1, 1, 1)$ and $d = (n, n, n)$ and give a strategy for the cops.

Claim 1. *Cop 1 can reach position with $c_3 = r_3$ within the first n rounds.*

The strategy of Cop 1 is to reach a position given by Claim 1, while Cop 2 stays put. Note that this takes at most n rounds, and from now on whenever the robber moves in z-dimension, Cop 1 maintains $c_3 = r_3$.

Claim 2. *Assuming $c_3 = r_3$, if the robber makes at least $n_1 = (n-1)+(2n-2)$ moves along the x- or y-dimension, then Cop 1 will catch him.*

Proof. We prove the claim in two steps. First we show that Cop 1 can reach a position with $c_1 - c_2 = r_1 - r_2$ in at most $n-1$ moves. If $c_1 - c_2 = r_1 - r_2$ at start, then we are done. So we may assume that $c_1 - c_2 < r_1 - r_2$. Cop 1 increases c_1. Then either $c_1 - c_2 = r_1 - r_2$, and we are done, or we still have $c_1 - c_2 < r_1 - r_2$. Then after the robber's move, we still have $c_1 - c_2 \leq r_1 - r_2$, and we can continue. If we never have equality then after $n - 1$ moves the cop is at position $(n, 1, 1)$, so $c_1 - c_2 = n - 1 \geq r_1 - r_2$. This contradicts $c_1 - c_2 < r_1 - r_2$.

Next, we show that if the robber makes additional $2n-2$ steps along x- or y-coordinate, he will be caught by Cop 1. Assume that $c_1 < r_1$ (and thus $c_2 < r_2$). When the robber increases r_1 or decreases r_2, then Cop 1 increases c_1 (which is possible since $c_1 < r_1 \leq n$). When the robber decreases r_1 or increases r_2, then Cop 1 increases c_2 (which is possible since $c_1 < r_1 \leq n$). Thus, in all cases he maintains the invariant $c_1 - c_2 = r_1 - r_2$. Moreover, the cop only increases his x- and y-coordinates, and he does so at most $2n-2$ times. Thus eventually $c_1 \geq r_1$. Since in each step at most one of c_1 and r_1 changes (by at most 1), at some point we must have $c_1 = r_1$, and thus $c_2 = r_2$. Since $c_3 = r_3$, this proves the claim. □

By Claim 1, we can assume that $c_3 = r_3$. At this point, Cop 2 changes his strategy such that he greedily moves towards the robber in x- and y-dimension.

Claim 3. *Assuming $c_3 = r_3$, Cop 2 can reach a position with $d_1 = r_1$ and $d_2 = r_2$ within $2n + n_1$ rounds.*

Proof. Whenever the robber moves along x- or y-dimension, Cop 1 follows the strategy from Claim 2 while Cop 2 maintains the difference $|d_1 - r_1| + |d_2 - r_2|$. In the case that the robber moves along z-dimension or stays put, Cop 2 moves along x- or y-dimension such that the difference $|d_1 - r_1| + |d_2 - r_2|$ decreases by one. Since in the next $2n + n_1$ rounds there are at least $2n$ moves when the robber doesn't move in x- or y- dimension (otherwise he will be caught by Cop 1) and $|d_1 - r_1| + |d_2 - r_2| \leq 2n$, we arrive in the position where $|d_1 - r_1| + |d_2 - r_2| = 0$. □

By Claim 1 and Claim 3, we can assume that after at most $n + 2n + n_1$ rounds we are in a situation with $c_3 = r_3$, $d_1 = r_1$ and $d_2 = r_2$. Cops strategy is now simple – if the robber moves along x- or y-dimension then Cop 1 follows the strategy from Claim 2 while Cop 2 maintains $d_1 = r_1$ and $d_2 = r_2$, and otherwise Cop 1 maintains $c_3 = r_3$ and Cop 2 decreases d_3 (assuming $r_3 < d_3$). If there are at least n_1 moves of the robber along x- or y-dimension, then by Claim 2 he will be caught by Cop 1. Therefore there has to be at least n moves where the robber either stays put or moves along the z-dimension, in which case he is caught by Cop 2.

Bonus question: what is the minimal number of cops needed to catch the robber in a k-dimensional grid?