LTCC Course: Graph Theory 2019/20
§2: Graphs on Surfaces; Graph Minors

Peter Allen

20 January 2020

Chapter 4 of Diestel is good for planar graphs, and Section 1.7 covers the notions of minor and topological minor. Section V.3 of Bollobás covers graphs on surfaces, and colourings thereof.

The definitive textbook for graphs on surfaces is: "Graphs on Surfaces", by Bojan Mohar and Carsten Thomassen, Johns Hopkins University Press, 2001; `www.fmf.uni-lj.si/~mohar/Book.html`

Most of what we will be discussing in the lectures and notes regarding graph minors can also be found in Chapter 12 of Diestel.

We continue to assume that our graphs are finite and simple (no loops or multiple edges). Much of the material could be adapted for non-simple graphs, but many things will go horribly wrong if we allow our graphs to be infinite.

It is also convenient to assume throughout that a graph has at least one vertex. (To paraphrase Frank Harary: "A graph without vertices is a pointless concept.")

Surfaces and Embeddings

A *(closed) surface* is a compact connected 2-manifold (i.e., every point has a neighbourhood homeomorphic to the open disc in $\mathbb{R}^2$). Surfaces can be classified as *orientable* and *non-orientable.* Moreover, each orientable surface is homeomorphic to one of the surfaces S_k, $k \geq 0$, where S_k is a "sphere with k handles". The sphere itself is S_0; the torus is S_1.

The surface S_k, for $k \geq 1$, can be constructed as follows. Take a convex region in the plane whose boundary is a $4k$-gon. Label the boundary segments consecutively as

$$\overrightarrow{a_1},\ \overrightarrow{b_1},\ \overleftarrow{a_1},\ \overleftarrow{b_1},\ \overrightarrow{a_2}, \overrightarrow{b_2},\ \overleftarrow{a_2},\ \overleftarrow{b_2},\ \ldots,\ \overrightarrow{a_k},\ \overrightarrow{b_k},\ \overleftarrow{a_k},\ \overleftarrow{b_k}.$$

Now identify the pairs of segments labelled $\overrightarrow{a_i}$ and $\overleftarrow{a_i}$, for each i, preserving the orientations given by the arrows, and do likewise for the $\overrightarrow{b_i}$ and $\overleftarrow{b_i}$. It's easy to see that this process identifies all the corners of the $4k$-gon into one point. (See Exercises.)

The *genus* of S_k is k, and its *Euler characteristic* χ is $2 - 2k$.

There is a similar construction giving all the non-orientable surfaces: N_k is formed from a $2k$-gon labelled as

$$\overrightarrow{a_1},\ \overrightarrow{a_1},\ \overrightarrow{a_2},\ \overrightarrow{a_2},\ \ldots,\ \overrightarrow{a_k}, \overrightarrow{a_k}.$$

The non-orientable surface N_k ($k \geq 1$) has genus k and Euler characteristic $2 - k$. The first two non-orientable surfaces in the list are the *projective plane* N_1 and the *Klein bottle* N_2.

More details can be found in Bollobás, for instance.

An *embedding* of a graph $G = (V, E)$ on a surface S is a function taking each vertex x of G to a point $\varphi(x)$ of S, and each edge xy of G to a *Jordan curve* in S, with endpoints $\varphi(x)$ and $\varphi(y)$, in such a way that the only intersections between the points and curves in the surface are those corresponding to incidences between edges and vertices of G. This all means what you think it ought to mean: this is exactly how we think of graphs being drawn on a surface.

A graph can be embedded on the sphere S_0 if and only if it can be embedded on the plane, in which case it is called a *planar graph*.

We'll skip the work required to develop enough machinery to prove anything rigorously. (E.g., the "Jordan curve theorem" says that a closed Jordan curve in the plane has an inside and an outside, and it's not easy to prove.) The result we need is that, if we remove the image of G from the surface S, we are left with a number of connected components called the *faces* of the embedding. The embedding is a *2-cell embedding* if each face is homeomorphic to the open unit disc.

In the case of the sphere S_0, the only way an embedding of a connected graph can fail to be a 2-cell embedding is if the graph has no vertices. For surfaces with more interesting topology, this is a non-trivial condition.

Two central questions of the subject are: (i) Given a surface S, which graphs can be embedded on S? (ii) Given a graph embedded on S, what can we say about its chromatic number?

We discussed (ii) a bit in the previous lecture. Here, we'll concentrate on (i).

The Euler-Poincaré Formula

The Euler-Poincaré Formula states that, if we have a 2-cell embedding of a graph on a surface S, then

$$v - e + f \; = \; \chi,$$

where v is the number of vertices of the graph, e is the number of edges of the graph, f is the number of faces of the embedding, and χ is the Euler characteristic of the surface S.

We'll just prove this in the case where S is the plane, whose Euler characteristic is 2.

Theorem 1 (Euler's Formula)**.** *Let G be a connected graph with at least one vertex, embedded in the plane. Then $v - e + f = 2$, where $v = |V(G)|$, $e = |E(G)|$, and f is the number of faces of the embedding.*

Proof. We work by induction on the number f of faces. When $f = 1$, the graph has no cycles, so is a tree, and $v = e + 1$, which is consistent with the formula.

For $f \geq 2$, we suppose the result is true for embeddings with at most $f - 1$ faces, and take an embedding of a graph with f faces. Choose an edge separating two different faces, and delete it. The graph remains connected: the number of faces has decreased by one, as has the number of edges, while the number of vertices is unchanged. By the induction hypothesis, Euler's Formula holds for the new embedding. Thus it holds for our embedding. Thus, by induction, the formula is valid for all embeddings. □

Euler's Formula is often quoted as referring to the number of vertices, edges and faces of a *convex polyhedron* in 3-space. The formula for polyhedra follows from the theorem for graphs, as a convex polyhedron can be "drawn in the plane" so that the notions of vertex, edge and face are preserved.

Euler's Formula is often used in conjunction with a "double-counting" of the edges in an embedding. For a face f which is homeomorphic to an open disc, let the *degree* $d(f)$ be the number of edges we encounter on the boundary walk of f. Note that if an edge appears twice in such a walk, we count it double. This means, for instance, that an embedding of an n-vertex tree in the plane yields one face with degree $2n-2$.

If F is the set of faces of the embedding, then we note that $\sum_{f\in F} d(f)$ counts the total number of edges on the boundaries of all the faces, and that each edge is counted exactly twice by this sum. So we have

$$\sum_{f\in F} d(f) \;=\; 2e.$$

Since all our graphs are finite, each face has at least 3 edges on its boundary, and hence in particular we see that $2e \geq 3f$. So using Euler's Formula now gives that for a planar graph we have

$$e \;\leq\; 3v-6.$$

Notice that this bound makes no mention of the embedding, so it gives a *necessary* condition for a graph to be planar. (Indeed, the same argument gives an upper bound on the number of edges of a graph that can be embedded on any given surface.)

By the Handshaking Lemma, we know that for any graph G we have $\sum_{v\in V(G)} d(v) = 2e$. Combining that with the formula above means $\sum_{v\in V(G)} d(v) \leq 6v-12$. This means that the average degree of any planar graph is strictly less than 6, so that any planar graph contains a vertex of degree at most 5. Hence $\deg(G) \leq 5$ for any planar graph, which implies that $\chi(G) \leq \mathrm{ch}(G) \leq 6$ – we saw last week that this can be improved!

Today, we head in a different direction. From the above inequality, we see that a planar graph on 5 vertices has at most 9 edges. This means that the complete graph K_5 is not planar.

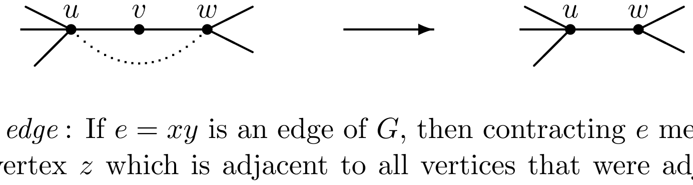
Also the complete bipartite graph $K_{3,3}$ is not planar. To see this, notice that, in any embedding of a bipartite graph on a surface, all faces have an even number of sides, so in particular at have degree at least 4. Thus we have $2e = \sum_{f\in F} d(f) \geq 4f$. Combining that with Euler's Formula leads to $e \leq 2v-4$, and this is false for $K_{3,3}$.

Subgraphs and minors

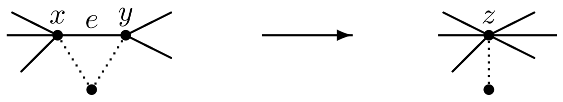
Now we know that K_5 and $K_{3,3}$ are not planar, we can deduce that any graphs "containing" them are not planar. For sure, this is true if our notion of containment is containment as a subgraph, but in fact we can make stronger statements by introducing more general notions of containment.

Let G be a graph. We define the following operations:

- *Removing a vertex* means removing that vertex from the vertex set of G and also removing all edges that vertices is incident with from the edge set.
- *Removing an edge* means removing that edge from the edge set of G.
- *Suppressing a vertex of degree two* means removing that vertex and adding an edge between its two neighbours, provided that edge is not already present (if the edge is already there, we don't add a new one).



- *Contracting an edge*: If $e = xy$ is an edge of G, then contracting e means removing x and y, adding a new vertex z which is adjacent to all vertices that were adjacent to x or y, after which multiple edges are removed.



Let H and G be two graphs.

- *H is an induced subgraph of G*, or *G has H as an induced subgraph*, notation $H \leq_I G$, if H can be obtained from G by a sequence of vertex removals.
- *H is a subgraph of G*, or *G has H as a subgraph*, notation $H \leq_S G$, if H can be obtained from G by a sequence of vertex and edge removals.
- *H is a topological minor of G*, or *G has H as a topological minor*, notation $H \leq_T G$, if H can be obtained from G by a sequence of vertex removals, edge removals, and suppression of vertices of degree two.
- *H is a minor of G*, or *G has H as a minor*, notation $H \leq_M G$, if H can be obtained from G by a sequence of vertex removals, edge removals, and edge contractions.

Note that in the definitions above we allow the sequences to have length zero, so every graph is a subgraph, etc., of itself.

There is a clear hierarchy of the order relations above:

$$H \leq_I G \quad\Longrightarrow\quad H \leq_S G \quad\Longrightarrow\quad H \leq_T G \quad\Longrightarrow\quad H \leq_M G.$$

The following useful result, whose proof is an exercise, gives an alternative characterisation of the minor relation.

Theorem 2. *The following two statements are equivalent for all graphs H, G:*

- *H is a minor of G.*
- *For each $u \in V(H)$, there exists a subset $V_u \subseteq V(G)$ of vertices from G so that*
  - *the sets $\{V_u \mid u \in V(H)\}$ are disjoint,*
  - *each set V_u, $u \in V(H)$, induces a connected subgraph of G, and*
  - *for all $u, v \in V(H)$ with $uv \in E(H)$, there are vertices $x \in V_u$, $y \in V_v$ with $xy \in E(G)$.*

Minors and Embeddings

Suppose that G is a planar graph, and that H is obtained from G by any of the operations of: vertex removal, edge removal, suppression of a vertex of degree 2, and edge contraction. We claim that H is also planar.

The first two of these are obvious. For suppression of vertices of degree 2, we obtain an embedding of H by replacing the two Jordan curves representing the edges removed from G by a single Jordan curve representing the new edge of H. The same also holds if we replace "planar" by "embeddable on the surface S", for any S.

For edge contraction, given an embedding of G, and an edge $e = xy$ of G to be contracted, we derive an embedding of H by placing the new vertex z anywhere on the curve representing xy, and extending all the curves incident with x or y inside thin tubes to reach z, following the path of the curve formerly representing xy.

The arguments above have the following consequence.

Theorem 3. *If G can be embedded on a surface S, and G contains H as a minor, then H can be embedded on S.*

We say that the family of graphs that can be embedded on a surface S is *minor-closed*: if G is in the family, and H is a minor of G, then H is in the family.

Because of the hierarchy of the operations introduced in the previous subsection, certainly if G can be embedded on a surface, and G contains H in any of the other senses discussed above, then H can be embedded on the surface.

Returning to the planar case, we now have the following results.

Theorem 4. *If G is planar, then:*

(1) G contains neither K_5 nor $K_{3,3}$ as a minor.

(2) G contains neither K_5 nor $K_{3,3}$ as a topological minor.

Kuratowski's Theorem says that the converses of both *(1)* and *(2)* in the previous theorem are true.

Theorem 5 (Kuratowski, 1930). *The following statements are equivalent for all graphs G:*

(a) G is planar.

(b) G contains neither K_5 nor $K_{3,3}$ as a minor.

(c) G contains neither K_5 nor $K_{3,3}$ as a topological minor.

We've seen that *(a)* implies *(b)*, and *(b)* implies *(c)*, since if G contains one of the graphs as a topological minor, it contains that graph as a minor.

This section contains only a proof that *(c)* implies *(b)*: if G contains one of K_5 or $K_{3,3}$ as a minor, then G contains one of K_5 or $K_{3,3}$ as a topological minor. It is important to stress that this is *not* a special case of any general result; it's more in the nature of a lucky accident that having one of K_5 and $K_{3,3}$ as a minor implies having one of the two as a topological minor.

Proof of *(c)* $\Rightarrow$ *(b)*. We use the characterisation of the graph minor relation given in Theorem 2.

Suppose first that G contains $K_{3,3}$ as a minor, and take a collection of six sets $V_u \subseteq V(G)$ as in Theorem 2, one for each $u \in V(K_{3,3})$. For each set V_u, we identify three edges to V_u from the sets V_w, where w is in the opposite class of $K_{3,3}$ from u. These "land" at three, not necessarily distinct, vertices of V_u: call these x, y, z. It is not hard to see that there is some vertex v_u in V_u (possibly equal to one or more of x, y, z) which has disjoint paths to x, y, z (possibly trivial, i.e., of length zero) in $G[V_u]$. (By $G[V_u]$ we denote the subgraph of G induced by the vertex set $V_u \subseteq V(G)$.) The six vertices w_u, $u \in V(K_{3,3})$, together with the various edges and paths we identified above, form a copy of a graph inside G that contains $K_{3,3}$ as a topological minor.

The paragraph above actually shows that if a graph G contains $K_{3,3}$ as a minor, then G contains $K_{3,3}$ as a topological minor. However – and hopefully this gives some insight into how and why the notions of minor and topological minor are different – if G contains K_5 as a minor, then it need not contain K_5 as a topological minor. Indeed, G can have a K_5 minor even if it has maximum degree 3, but a graph with K_5 as a topological minor must have at least five vertices of degree 4.

To complete the proof of *(c)* $\Rightarrow$ *(b)*, we need to prove that, if G contains K_5 as a minor, then it contains either K_5 or $K_{3,3}$ as a topological minor. Again we use the characterisation from Theorem 2. So suppose that there are five disjoint connected sets V_a, V_b, V_c, V_d, V_e in G, with edges between each pair. The plan is to set off trying to find K_5 as a topological minor. So, for each

V_i, $i \in \{a,b,c,d,e\}$, we find the four "landing points" $x_{ij} \in V_i$ of the edges that connect V_i to the other V_j. Either there is a vertex w_i in V_i with four disjoint paths to the x_{ij}, or there are two vertices f and g in V_i, connected by a path, with two of the x_{ij} sending paths to f and the other two to g, all five paths being internally disjoint. If the first case occurs for all V_i, then we found a topological minor of K_5 in G. If the latter case occurs with any of the V_i, then we change plans: in this case, we divide V_i into two connected parts, one containing f and two of the x_{ij}, and the other containing g and the other two x_{ij} – the six vertex sets now witness that $K_{3,3}$ is also a minor, and therefore a topological minor, of G. □

Of course, the hard part of Theorem 5 is to prove that (*b*) and (*c*) imply (*a*); given a graph G which doesn't contain K_5 or $K_{3,3}$ as a minor, we want to find a plane embedding of G.

We give only a brief sketch. The proof goes by induction on $v(G)$. If G is not connected, then we draw its connected components (by induction) disjointly in the plane.[1] If G is connected but not 2-connected, we can do something similar: let v be a vertex such that $G - v$ is disconnected, and let C be a component of $G - v$. We can draw the graphs $G - C$ and $G\big[C \cup \{v\}\big]$ in the plane by induction; we put the drawing of the second, without loss of generality with v on the outer face, in a face of the drawing of the first containing v, and identify the vertex v in the two drawings.

To deal with 2-connected but not 3-connected G becomes a little harder. Suppose $\{u, v\}$ is a separating set, that is $G - \{u, v\}$ is disconnected, and let C be a component of $G - \{u, v\}$; let $G_1 = G - C$ and $G_2 = G\big[C \cup \{u, v\}\big]$. If uv is an edge of G, then we can repeat the trick of drawing G_1 and G_2 by induction and putting the drawing of G_2, with uv on an outer face, in a face of G_1 containing uv, then identifying u and v. If uv is not an edge of G, this doesn't work—but the trick is to notice that we can simply add uv to both G_1 and G_2 without creating a K_5 or $K_{3,3}$ topological minor, then use induction. Suppose $G_1 \cup \{uv\}$ has one of these two graphs as a topological minor. The edge uv must be used in this topological minor, since G_1 does not contain $K_{3,3}$ or K_5 as a topological minor. But there is a path in G_2 from u to v: for if not, then $G - u$ would be disconnected. We can replace uv with this path to find $K_{3,3}$ or K_5 as a topological minor in G, a contradiction.

Finally, we can assume G is 3-connected. This is the point where we really need to do some work, and where I will skip most of the details. Note that this sketch is substantially appealing to your intuition on how planar drawings work; if you were forced to actually write down details of how to identify the two copies of uv given planar drawings, it would be rather painful (and long). We can assume that all edges in all our planar drawings are straight lines (this property would easily be preserved by the induction steps above, and the following sketch gives it for the 3-connected case) which makes life rather easier, but still not really easy.

Briefly, to deal with the case G is 3-connected, either we have $G = K_4$ and it is easy to draw a plane embedding (with straight lines) or G has 5 or more vertices. One can fairly easily check that in this case (Lemma 3.2.4 in Diestel) there is some edge xy such that when we contract xy we obtain a graph G' which is still 3-connected. We can draw this in the plane by induction; let z be the vertex obtained by contracting xy. Let C be the face of $G' - z$ containing z. We try to draw x and y inside C, with a straight line joining them, and add more straight lines to form a drawing of C. Now by a case analysis (Lemma 4.4.3 of Diestel) you can check that either this is possible, or you find a K_5 or $K_{3,3}$ minor.

[1]Note that this works for the plane; it would not work for the torus or other more complicated surfaces.

Orderings and closedness of properties

We will now leave the topic of graphs on surfaces, and examine the notions of graph containment for their own sake. To begin with, we observe that our relations of containment are all *transitive* (if G contains H and H contains J, then G contains J), and so give "orderings" on the set of all graphs: let us be more precise.

If $\preccurlyeq$ is a relation on a set X, then $(X, \preccurlyeq)$ is called a *quasi-ordering* or *pre-order* if the relation is reflexive ($x \preccurlyeq x$ for all $x \in X$) and transitive ($(x \preccurlyeq y \wedge y \preccurlyeq z) \Rightarrow (x \preccurlyeq z)$ for all $x, y, z \in X$).

We say that a quasi-ordering $(X, \preccurlyeq)$ is *without infinite descent* if there is no infinite strictly decreasing sequence $x_1 \succ x_2 \succ x_3 \succ \cdots$ (where $x \succ y$ means $y \preccurlyeq x$ and $x \neq y$).

It is easy to see that the orderings defined in the previous section on the class $\mathscr{G}$ of all (simple, finite) graphs correspond to quasi-orderings without infinite descent.

A subset $A \subseteq X$ of a quasi-ordering $(X, \preccurlyeq)$ is an *antichain* if every two elements from A are incomparable (i.e., if $a, b \in A$ with $a \neq b$, then $a \not\preccurlyeq b$ and $b \not\preccurlyeq a$).

Proposition 6. *Given a quasi-ordering $(X, \preccurlyeq)$, any infinite sequence $a_1, \ldots$ in X either repeats one term infinitely often or contains at least one of the following three things: an infinite strictly decreasing sequence, an infinite strictly increasing sequence, and an infinite antichain.*

Proof. This is really a special case of the infinite Ramsey theorem—let's prove it in that language. We colour the complete graph on $\mathbb{N}$ with four colours, colouring the edge ij (for $i < j$) with the relation between a_i and a_j (which can be one of $a_i = a_j$, or $a_i \prec a_j$, or $a_j \prec a_i$, or none of the above). We claim that for any 4-colouring of the complete graph on $\mathbb{N}$ there is an infinite set S such that all pairs in S have the same colour; the existence of this set proves the proposition.

To find such a set, we work as follows. Let $s_1 = 1$, and colour s_1 with a colour c_1 appearing on infinitely many edges leaving 1; let T_1 be the set of vertices joined to s_1 with colour c_1. Now for each $i = 2, \ldots$, let s_i be the smallest element of T_{i-1}. Let c_i be a colour appearing on infinitely many edges from s_i to T_{i-1}, and let T_i be the set of vertices in T_{i-1} joined to s_i with colour c_i. We obtain in this way an infinite sequence since by construction each T_i is infinite.

Now pick a colour c which appears infinitely often in the sequence $c_1, \ldots$, and let $S = \{s_i : c_i = c\}$. It is easy to check that S is the desired infinite monochromatic clique. Note that this proof doesn't rely on having specifically 4 colours; any finite number would work. ☐

Let P be a property defined on the elements of X. We say that P is *closed under* $\preccurlyeq$ or $\preccurlyeq$*-closed* if for every two elements $x, y \in X$ we have that if x has property P and $y \preccurlyeq x$, then y also has property P.

As an example, suppose property P is defined for $G \in \mathscr{G}$ (the set of all graphs) as "G is bipartite". This property is closed under both the subgraph and the induced subgraph ordering, but not under the topological minor or the minor ordering. (See Exercises.)

Let $(X, \preccurlyeq)$ be a quasi-ordering and suppose P is a $\preccurlyeq$-closed property defined on the elements of X. Then we can talk about the set $\mathscr{P}$ of all elements in X that satisfy property P. And of course we also have the complement $\overline{\mathscr{P}} = X \setminus \mathscr{P}$ of all elements in X that do *not* satisfy property P. Let M be a set of minimal elements of $\overline{\mathscr{P}}$

Since P is assumed to be $\preccurlyeq$-closed, we know that if $x \in \overline{\mathscr{P}}$ and $x \preccurlyeq y$, then $y \in \overline{\mathscr{P}}$. This leads to the following crucial observation :

$$x \textit{ has property } P \qquad \Longleftrightarrow \qquad \textit{there is no } m \in M \textit{ with } m \preccurlyeq x.$$

In other words : a property that is $\preccurlyeq$-closed is completely determined once we know a set of minimal elements of the set of elements that don't have the property. Such a minimal set is called a *minimal forbidden set* of the property.

The observations above *may* provide a good description of certain properties and *may* provide efficient algorithms to test if a given element satisfies the property. This possible usefulness depends on the answers to questions like : Can we find a minimal forbidden set ? Is this set finite ? Is there a good algorithm to test if $x \preccurlyeq y$ or not ? Etc.

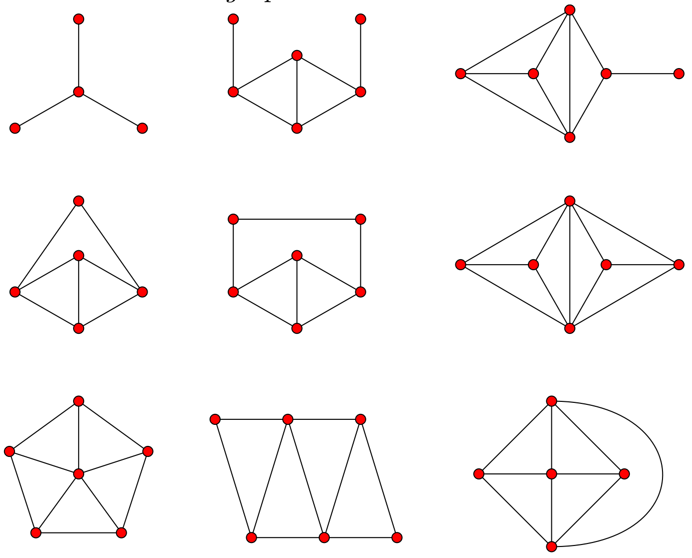
You may wonder why we look at a set of minimal elements of the set $\overline{\mathscr{P}}$ of elements in X that do *not* satisfy property P. Wouldn't it be more natural to look at the set of maximal elements of $\mathscr{P}$? Yes, it would be more natural. But for the orderings we are considering such a set of maximal elements usually doesn't exist. The orderings give natural minimal elements, since from every finite graph, we can only have a finite number of descending steps before we have to stop (we've reached a graph with one vertex, say). But in general we won't have maximal elements (except for very special properties).

Here is an example illustrating the concepts above.

For a graph $G = (V, E)$, recall that the *line graph* $L(G) = (V_L, E_L)$ is the graph that has the edges of G as vertices : $V_L = E$; and two edges are adjacent in the line graph if they have a common end-vertex in G. A graph H is a line graph if $H \cong L(G)$ for some graph G.

It's easy to see that if H is a line graph, then every induced subgraph of H is also a line graph. Hence the property of "being a line graph", defined on the set $\mathscr{G}$ of graphs, is closed under the induced subgraph ordering $\leq_I$. For this property, we actually do know the unique set of minimal forbidden elements.

Theorem 7 (Beineke, 1968). *A graph H is a line graph if and only if it does not contain one of the nine graphs below as an induced subgraph.*



Well-quasi-ordering

A quasi-ordering $(X, \preccurlyeq)$ is a *well-quasi-ordering* if for every infinite sequence $x_1, x_2, \ldots$ of elements from X, there are two indices $i < j$ so that $x_i \preccurlyeq x_j$. In particular, Proposition 6 implies the following.

Proposition 8. *The following two properties are equivalent for a quasi-ordering $(X, \preccurlyeq)$:*

- $(X, \preccurlyeq)$ *is a well-quasi-ordering;*
- $(X, \preccurlyeq)$ *is a quasi-ordering without infinite descent and without infinite antichains.* □

Requiring a quasi-ordering to be a well-quasi-ordering is a very strong requirement. For instance, for the graph orderings defined in the first section, neither $(\mathscr{G}, \leq_I)$, nor $(\mathscr{G}, \leq_S)$, nor $(\mathscr{G}, \leq_T)$, are well-quasi-orderings. For the induced subgraph ordering and the subgraph ordering, the sequence of cycles $C_3, C_4, C_5, \dots$ forms an infinite sequence that fails the condition in the definition. In one of the exercises you will be asked to find counterexamples yourself for the topological minor ordering.

A fundamental property of well-quasi-orderings is that given a well-quasi-ordering $(X, \preccurlyeq)$, one can build a new well-quasi-ordering as follows. Let $X^{<\infty}$ denote the set of finite subsets of X. Given $A, B \in X^{<\infty}$, we write $A \preccurlyeq B$ if there is an injection $f : A \to B$ such that $a \preccurlyeq f(a)$ for each $a \in A$. This turns out to be a well-quasi-order.

Lemma 9 (Higman's Lemma, 1952)**.** *The quasi-ordering $(X^{<\infty}, \preccurlyeq)$ is a well-quasi-ordering.*

Proof. It's easy to check that since $(X, \preccurlyeq)$ is a quasi-ordering, so is $(X^{<\infty}, \preccurlyeq)$. So what we need to show is that there is no infinite descending sequence or antichain; in other words, for any sequence $A_1, \dots$ in $X^{<\infty}$, there are $i < j$ such that $A_i \preccurlyeq A_j$. Call a sequence with this property *good*, and one which does not have it *bad*. Suppose for contradiction that such bad sequences exist.

We now construct a sequence. First, let A_1 be an element of $X^{<\infty}$ with fewest elements such that A_1 is the first term of some bad sequence. Now, for each $i = 2, \dots,$ given $A_1, \dots, A_{i-1}$, let A_i be an element of $X^{<\infty}$ with fewest elements such that $A_1, \dots, A_i$ is an initial segment of some bad sequence. By construction, the result is a bad sequence.

Because $\emptyset \preccurlyeq A$ for each $A \in X^{<\infty}$, we have $|A_i| \geq 1$ for each i. Thus for each i we can choose $a_i \in A_i$. Let $B_i = A_i \setminus \{a_i\}$ for each i. Because $(X, \preccurlyeq)$ is a well-quasi-ordering, by Proposition 6 there is an infinite (not necessarily strictly) increasing subsequence $(a_{i_1}, \dots)$ of $(a_i)_{i=1}^{\infty}$. Now consider the sequence $A_1, \dots, A_{i_1-1}, B_{i_1}, B_{i_2}, \dots$. By choice of A_{i_1} this sequence is good. We do not have $A_i \preccurlyeq A_j$ for any $i < j$, and since $B_j \preccurlyeq A_j$ using the trivial identity injection, we also do not have $A_i \preccurlyeq B_j$ for any $i < j$. Thus the reason that $A_1, \dots, A_{i_1-1}, B_{i_1}, B_{i_2}, \dots$ is good is that there exist k and ℓ with $k < \ell$ such that $B_{i_k} \preccurlyeq B_{i_\ell}$. Let $\phi : B_{i_k} \to B_{i_\ell}$ be an injection witnessing $B_{i_k} \preccurlyeq B_{i_\ell}$, and let ϕ' be the injection from A_{i_k} to A_{i_ℓ} extending ϕ with $\phi'(a_{i_k}) = a_{i_\ell}$. Now ϕ' witnesses $A_{i_k} \preccurlyeq A_{i_\ell}$, a contradiction. □

Although the whole class of graphs is not well-quasi-ordered under the topological minor ordering, some important subclasses are.

Theorem 10 (Kruskal, 1960)**.** *The class of all trees, with topological minor as the ordering, is well-quasi-ordered.*

The proof of this is quite similar to that of Lemma 9, with one extra trick.

Proof. We aim to prove a slightly stronger statement. A *rooted tree* is a tree T with one vertex identified as the *root*. Now a tree T is a topological minor of another tree T' if and only if we can subdivide the edges of T (that is, we are allowed to perform the operation of replacing an edge uv with a new vertex w and the edges uw, vw as often as we like) and find an isomorphism ϕ from the result to a subgraph of T'. If T and T' are rooted trees, we say that in addition ϕ is *order-preserving* if the following condition holds. For any $x, y \in V(T)$ such that the path from the root of T to y goes through x, so the path from the root of T' to $\phi(y)$ goes through $\phi(x)$. We will prove that the class $\mathcal{T}$ of all rooted trees, with order-preserving topological minor as the ordering, is well-quasi-ordered. We denote this ordering by $\preccurlyeq_O$.

Let us now show that no bad sequence (as in Lemma 9) of rooted trees exists. Suppose to the contrary that such a bad sequence exists. We construct a sequence $T_1, T_2, \ldots$ inductively as follows (essentially as in the proof of Lemma 9). Let T_1 be a smallest (number of vertices) rooted tree which is in such a bad sequence. Now for each $n \geq 1$ successively, given $T_1, \ldots, T_n$, choose T_{n+1} to be a smallest rooted tree such that $T_1, \ldots, T_{n+1}$ is an initial segment of a bad sequence. Let $r_1, \ldots$ be the roots of the trees $T_1, \ldots$. Note that trivially each T_i has at least two vertices (actually at least three is also trivial; but we won't need it).

For each i, observe that $T_i - r_i$ forms a finite collection of trees, which we root at the vertices adjacent to r_i in T_i. For each i, let A_i be the set of rooted trees $T_i - r_i$. Now our goal is to show that for some $i < j$ we have $A_i \preccurlyeq_O A_j$, where as with Lemma 9 we derive a quasi-ordering on $\mathcal{T}^{<\infty}$ from that on $\mathcal{T}$. Given such a pair, since $A_i \preccurlyeq_O A_j$ we have a collection of maps witnessing this, whose union is a map from $T_i - r_i$ to $T_j - r_j$. Extending this map by adding $r_i \to r_j$ we have a map from T_i to T_j which witnesses $T_i \preccurlyeq_O T_j$; that is the contradiction we want.

If we knew $(\mathcal{T}, \preccurlyeq_O)$ were a well-quasi-ordering, then, by Lemma 9,$(\mathcal{T}^{<\infty}, \preccurlyeq_O)$ would be a well-quasi-ordering as well, which implies what we want. Of course, this is exactly what we are trying to prove. The trick is to show that the set of rooted trees $\mathcal{A} := \bigcup_i A_i$ is demonstrably well-quasi-ordered by $\preccurlyeq_O$. We get the desired existence of $A_i \preccurlyeq_O A_j$ by applying Lemma 9 to this set.

It remains to show that $(\mathcal{A}, \preccurlyeq_O)$ is a well-quasi-ordering. To that end, let $B_1, \ldots$ be a sequence of trees in $\mathcal{A}$. For each i, let $n(i)$ be such that $B_i \in A_{n(i)}$, and choose k minimising $n(k)$. Trivially B_k has fewer vertices than $T_{n(k)}$, so by construction of the sequence $T_1, \ldots$, it follows that the sequence $T_1, T_2, \ldots, T_{n(k)-1}, B_k, B_{k+1}, \ldots$ is good. Let $B \preccurlyeq_O B'$ be a pair witnessing that this sequence is good (so B comes before B' in the sequence). Now B cannot be in the initial segment $T_1, \ldots, T_{n(k)-1}$, since otherwise either B' is also in this initial segment, contradicting badness of $T_1, \ldots$, or $B' = B_i$ for some i, but then we have $B \preccurlyeq_O B' = B_i \preccurlyeq_O T_{n(i)}$ and since $n(i) > n(k) - 1$ by choice of k, this too is a contradiction to badness of $T_1, \ldots$. But then we have $B = B_i$ and $B' = B_j$ for some $i < j$; in other words, the sequence $B_1, \ldots$ is good, as desired. □

An interesting point to note here is that this proof (and that of Higman's Lemma 9) really use infinite sets and the Axiom of Choice in a fundamental way—not a very common thing in discrete mathematics. It's easy enough to see that in Theorem 10 we really need an infinite set to guarantee finding two trees where one is a topological minor of another (for any finite n we can just take all non-isomorphic n-vertex trees; the number of such trees tends to infinity with n). But Kruskal's theorem in particular implies the following, observed by Friedman.

Corollary 11. *Let* $\operatorname{tree}(n)$ *denote the minimum* m *such that the following statement holds. For every sequence* $T_1, \ldots, T_m$ *of trees, where* T_i *has* $i + n$ *vertices, there is some pair with one a topological minor of the other.*

Then $\operatorname{tree}(n)$ *exists.*

Proof. This is a classical *compactness argument.* Suppose $\operatorname{tree}(n)$ does not exist; then for each m there is a counterexample sequence of m trees. Let T_1 be a tree which plays the rôle of T_1 in infinitely many of these sequences, and discard the other sequences. Let T_2 be a tree which plays the rôle of T_2 in infinitely many of these sequences, and discard the other sequences. Repeat forever; the result is an infinite sequence of trees, which by Theorem 10 is good. Fix a pair T_i, T_j with $i < j$ witnessing this, and we see that the supposed counterexample sequence for $m = j$ contains the pair T_i, T_j where T_i is a topological minor of T_j. □

This proof provides no hint of how fast $\operatorname{tree}(n)$ grows. In fact, it turns out to grow incredibly fast—so fast that it cannot really be described by 'Peano arithmetic' (this is a minimal system of axioms that let you perform arithmetic; normally that's about all we need in discrete mathematics), so that while you can prove any given tree(n) exists in Peano arithmetic (by brute force enumeration, say) you cannot prove Corollary 11 in Peano arithmetic (at least, if ZFC is consistent). You really need to assume stronger axioms (like the Axiom of Choice) which normally aren't needed in discrete mathematics. This is also a result of Friedman (indeed, this motivated his observation of Corollary 11); the idea is to show that on the one hand Peano arithmetic cannot prove the existence of very fast-growing functions, and on the other hand that tree(n) does grow so fast.

We conclude this section with an easy but important proposition.

Proposition 12. *Let $(X, \preccurlyeq)$ be a well-quasi-ordering and P a $\preccurlyeq$-closed property on X. Then a minimal forbidden set of P is finite.*

Proof. Let $\mathcal{F}$ be a forbidden set of P. If $\mathcal{F}$ is infinite, then by definition there are $F_1 \neq F_2$ such that $F_1 \preccurlyeq F_2$. Let $\mathcal{F}' = \mathcal{F} - \{F_2\}$. We claim that $\mathcal{F}'$ is a forbidden set of P. Indeed, if $Q \notin P$, then by definition of $\mathcal{F}$ there exists $F \in \mathcal{F}$ such that $F \preccurlyeq Q$. Either $F \neq F_2$, in which case $F \in \mathcal{F}'$, or $F = F_2$, in which case we have $F_1 \preccurlyeq F_2 \preccurlyeq Q$ and hence $F_1 \preccurlyeq Q$. It follows that $\mathcal{F}$ is not a minimal forbidden set of P. □

The importance of this property is that once we know that $(X, \preccurlyeq)$ is a well-quasi-ordering, then *every* property that is $\preccurlyeq$-closed has a finite minimal forbidden set. *If* we also have, for each $x \in X$, an algorithm which efficiently answers the question 'Given $y \in X$, is $x \preccurlyeq y$?', then that proves that for any $\preccurlyeq$-closed property P, there exists an efficient algorithm that decides the question 'Given $y \in X$, is $y \in P$?'.

But note that the proofs we saw so far that some quasi-orderings are well-quasi-orderings are not constructive— in particular, they don't help us find these minimal forbidden sets guaranteed by Proposition 12. So when we can test $x \preccurlyeq y$ efficiently, we are in the rather strange position of knowing efficient algorithms exist without being able to find them. We'll see in the next section that this is exactly the case for minor-closed graph properties.

Minors of graphs

In the early 1980s, Robertson and Seymour announced a proof of a conjecture of Wagner: that the class of finite graphs with the minor ordering is a well-quasi-ordering. The proof spans 20 papers and over 400 pages which appeared over the next 20 years, the 'Graph Minors' series (and the series continued for a few more papers to tie up some related problems).

Theorem 13 (Graph Minor Theorem, Robertson & Seymour, 1986–2004)**.** *The class of finite graphs is well-quasi-ordered under the minor ordering.*

They also proved, in this series, that indeed, for any fixed G, one can check if G is a minor of an input graph H in time cubic in $|E(H)|$ (and the algorithm for this is completely explicit), so that for any given minor-closed class of graphs $\mathcal{G}$, one can decide in cubic time if $H \in \mathcal{G}$. But, unless you are given $\mathcal{G}$ in a way which makes it obvious what the list of forbidden minors is, then you probably cannot find this list in any useful way. Indeed, if $\mathcal{G}$ is given via Turing machines (which is a reasonable enough way) then the problem is known to be undecidable (Fellows and Langston, 1989). To take a more practical view, we know there are 2 forbidden minors for the class of planar graphs (Theorem 5). For the next orientable surface, the torus, there are over 239,000 known forbidden minors (found by computer search); and there is no reason to believe that the total number is close to this.

We can try to sketch, very briefly(!) how the Graph Minor Theorem is proved. For now, consider only planar graphs. Let $G_1, \dots$ be a sequence of planar graphs.

If this sequence contains as minors arbitrarily large 'grids', then it is a good sequence. A grid of size k is simply a graph drawn on the integer points $[k] \times [k]$ in the plane by joining each point to the at most four points at unit distance from it. A large grid 'obviously' contains any small planar graph (say G_1) as a minor: to see this, enlarge your (straight line) planar drawing of G_1 by a large factor, replace the edges and vertices with small rectangles and circles respectively, and superimpose it on the grid; this gives an 'obvious' minor.

On the other hand, the case that all the planar graphs 'look like a tree' is easy (at least by comparison to the rest of this monumental proof) to handle; one can start with Theorem 10 and try to generalise it.

To be more precise about what 'looks like a tree' means, we need to say what a *tree decomposition* of a graph G is. This is a tree T, whose nodes are sets of vertices of G, with the following two properties. First, every edge of G goes between two vertices which are in some node set of T. Second, every vertex of G is in a collection of node sets of T which form a subtree of T. The *width* of such a tree decomposition is defined to be the size of a largest node set minus one (the reason for subtracting 1 is just to look pretty: so that trees have tree decompositions of width 1). Finally, the *treewidth* of a graph G is the minimum width of any tree decomposition of G. This parameter turns out to be important in a lot of graph theory, especially when one starts doing algorithms. Here, the relevance is that if there is some r such that all graphs $G_1, \dots$ have treewidth at most r, then one can push through the generalisation of Theorem 10 to show that $G_1, \dots$ is a good sequence.

Finally, we are left to deal with sequences $G_1, \dots$ which don't contain arbitrarily large grid minors, but where there is no bound on the treewidth. Somewhat surprisingly (and the proof is not easy), Robertson and Seymour proved that there is no such sequence: for any k, there is r such that a graph whose treewidth exceeds r necessarily contains a grid of size k as a minor.

Moving to the general case, very vaguely, the idea is as follows. First, we can assume no graph in our sequence contains a K_t minor for $t = v(G_1)$. Then, we prove that any graph which doesn't contain a K_t minor has a tree-decomposition in which all the node-sets induce 'nice' graphs, and we prove that 'nice' graphs are well-quasi-ordered, and then we can use something like Kruskal's theorem to put the pieces together. Unfortunately the definition of 'nice' gets rather complicated for the general case; a short version is 'almost embeds in some surface that K_t does not embed into', but this short version omits so many technicalities it's almost a lie. In any case, the point is that there are only finitely many of these surfaces, and a sufficiently complicated graph on a surface S contains all small graphs on S as minors (by the same sort of argument as why a large grid contains all small planar graphs). Of course, this sketch too leaves out so many details it is almost closer to false than true.

Computation

Earlier, and last week, we mentioned algorithms a lot. Now we want to try to formalise what exactly 'an algorithm' to 'solve a problem' is. The simplest way to do this is to define a Turing Machine (TM).

Turing Machines

A (deterministic) TM has the following parts:

- an *alphabet* Σ, which is a finite set of symbols, one of which is #, called 'blank' (A modern computer uses two symbols, but it is not exactly a TM; if you want to work with TMs it is usually convenient to use a larger alphabet),

- a 2-*way infinite tape*, divided into *squares*, on each of which a symbol from Σ is written,
- a *tape head* which is always 'at' one square of the tape,
- a finite list Q of *states*, and
- a *transition function* $\delta : Q \times \Sigma \to Q \times \Sigma \times \{\text{Left}, \text{Right}, \text{Halt}\}$.

Let M be a TM. At time 0, we always assume that 'the state of M' is the initial state q_0. We call the square at which the tape head of M is initially positioned the 'zero square' (purely for convenience in discussion: the squares are *not* numbered in a way visible to M!). A finite string I, called the *input*, is written on the tape, starting at the zero square and moving right, and all other squares on the tape are blank (i.e. # is written on them).

Now for each time $t \geq 1$ in succession, the following happens. Suppose M is in state q at time $t-1$, and suppose that the symbol σ is written on the square at which the tape head is positioned at time $t-1$. Let $\delta(q, \sigma) = (q', \sigma', m)$. Then the state of M at time t is q' and the symbol σ' is written on the tape square where the tape head is positioned at time $t-1$ (overwriting σ). The tape is otherwise not changed from time $t-1$ to time t. Finally, the tape head is moved Left or Right according to m, or if $m = \text{Halt}$ then the TM M halts, meaning no further action ever occurs. Observe that by definition, at any given finite time the tape head has visited only finitely many tape squares, so all but finitely many squares of the tape are still blank.

It will sometimes (next lecture!) be useful to generalise this to a *multi-tape TM*, in which case we have a (finite) list of 2-way infinite tapes, each with a tape head which moves independently; the transition function reads all the tapes, updates all the tapes, and sets movement for each tape in each time step.

A good example of a TM is a modern computer with the hard disc replaced by a 2-way infinite tape (as above). Let's call this a *tape-computer*. The computer can do all kinds of complicated computation in between reading and writing the tape, but ultimately this computation takes only a finite time (because there are only so many possibilities for the internal memory—let's assume the computer avoids infinite loops in this internal computation!) and can be reduced to simply changing from one of a (very long!) list of states to another. You should believe from this example that a TM can do anything that a modern computer can do, if it has enough states.

In particular, one thing a tape-computer can do is *simulate* a TM, that is, if you input to a tape-computer a description of a TM M and its input I (written on the tape), then the tape-computer can work out for each time t what the result of running M on input I for time t will be. This makes it a *universal TM*; that's more or less the abstract definition of 'programmable computer'. So such things exist, if you use a TM with enough states. In fact, 'enough states' is not the colossal number that comes from this construction; 10 states is (easily) enough, though proving this is quite painful.

A given TM M with an input I can do one of two things: either it eventually halts, having *computed* the final contents of the tape (the finite non-blank string of symbols), or it does not halt in finite time.

Problems

The standard way to describe a problem in theoretical computer science is as follows:

NAME-OF-PROBLEM-IN-CAPITALS
Input : *A description of the type of inputs that will be considered.*
Output : *Certain kind of output, usually related to the input.*

An example is :

INTEGER-ADDITION
Input : Two integers a and b.
Output : The sum $a + b$ of a and b.

An *instance* of a problem is a specific input of the prescribed type. So, an instance for INTEGER-ADDITION could be "$a = 514$ and $b = -6969696969$".

In fact, the above is more general than what we will study in these lectures. We will mostly look at so-called *decision problems.* These are problems in which the output should only be "Yes" or "No", depending on the outcome of a certain question.

Examples would be :

GRAPH-COLOURING
Input : An undirected graph G and an integer $K \geq 1$.
Question : Does there exist a vertex colouring of G with K colours ?

GRAPH-K-COLOURING
Input : An undirected graph G.
Question : Does there exist a vertex colouring of G with K colours ?

Instead of "an instance for which the answer is "Yes"", we will say "a true instance".

Note that there really is a difference between the decision problems GRAPH-COLOURING and GRAPH-K-COLOURING. The latter one only makes sense if the value of K is known or given from the outset. Once the value of K is decided or given, that K is used for every instance. In the first problem, K is part of the input and can vary from instance to instance.

It is important not to confuse the 'mathematical difficulty' of a problem with the 'algorithmic difficulty'. For example, INTEGER-ADDITION is mathematically 'trivial', but making a machine which solves it, while not hard, is not trivial. On the other hand, consider the decision problem

FLT
Input : An integer $n \geq 1$.
Question : Does there exist a solution to $x^n + y^n = z^n$ with $x, y, z \geq 1$ integers ?

It's easy to see that the answer to this problem is 'Yes' if $n = 1$ or $n = 2$, but it is a famous and very difficult theorem of Wiles ('Fermat's Last Theorem') that for all larger n the answer is 'No' – so this problem is mathematically very far from trivial. Nevertheless, a machine which solves the problem is trivial to build – it must simply return 'Yes' if $n = 1, 2$ and otherwise 'No'. In fact (for many reasonable ways of inputting n) the machine doesn't even have to read the whole input and stops in constant time!

Words and Languages

Given an alphabet Σ (which we will assume is fixed from now on), a *word* is a finite sequence of symbols from the alphabet. The *length of a word x*, denoted $|x|$, is the number of symbols in it.
A special word is the *empty word*, often denoted by Λ. The empty word plays a role comparable to the empty set. In particular we have $|\Lambda| = 0$.

Given an alphabet Σ, we denote by Σ^* the set of all words using symbols from Σ. So we would have $\{2, x\}^* = \big\{\Lambda,\ 2,\ x,\ 22,\ 2x,\ x2,\ xx,\ 222,\ \ldots\big\}$.

Given an alphabet Σ, a *language* $\mathcal{L}$ is a subset of Σ^* : $\mathcal{L} \subseteq \Sigma^*$.

In these lectures we don't want to spend time worrying about how to give the input to a machine. So we ignore most of the questions regarding "encoding" of instances in words over a certain alphabet. We just expect that it is possible for the problems we encounter; that there is some "natural encoding" of the instances of the problem we are looking at.

We require one property of any encoding of the instances of a problem Π: there can be no words $x \in \Sigma^*$ that encode more than one instance of Π.

The above description fits our experience with computers. Almost all modern computers do everything using just $\{0, 1\}$ as the alphabet. And even with that restricted alphabet there seem to be few restrictions of what can be represented to a modern computer.
(Actually, there are major restrictions on what computers can deal with. In particular they can't work with binary expansions of real numbers like $\sqrt{2}$ and π.)

The only explicit encoding we will use is that if $\Sigma = \{0, 1\}$, then a non-negative integer N will be encoded as a binary number of length $\lfloor \log_2(N) \rfloor + 1$.
All logarithms from now on will be assumed to be with base 2.

With the above convention on encoding in mind, we can define for a decision problem Π the corresponding language $\mathcal{L}_\Pi$:

$$\mathcal{L}_\Pi = \bigl\{ x \in \Sigma^* \mid x \text{ is an encoding of a true instance of } \Pi \bigr\}.$$

And in fact, we will more or less identify the problem Π and the language $\mathcal{L}_\Pi$.

Despite the fact that we are glossing over 'encoding issues' in this course, you should be aware that they are sometimes very important. A good example is that it is possible to encode an integer either in 'unary' (n is represented by a string of n ones) or 'binary' (with which you are familiar), or indeed in many other ways. Consider the following two problems:

FACTORISE-UNARY
Input: An integer $n \geq 1$ presented in unary form.
Output: The prime factors of n in increasing order.

FACTORISE-BINARY
Input: An integer $n \geq 1$ presented in binary form.
Output: The prime factors of n in increasing order.

Now the first problem is easy to solve in polynomial time (polynomial in the length of the input), while the second is widely believed to be impossible to solve quickly—and any quick algorithm would leave the world's private communication and banking wide open to criminals by making it easy to break the RSA system. The difference is that when n is input to the first problem, the length of the input is n, whereas when it is input to the second problem, the length of the input is $\lceil \log_2 n \rceil$. So in the first problem we are 'allowed' to take a much longer time to factorise n than in the second. Similarly (although this is more obviously 'cheating') we could define a representation of a graph in which we require that the vertices are given in colour order from some optimal proper vertex colouring. The problem CHEATING-GRAPH-COLOURING, which takes this representation together with an integer $K \geq 1$ as input and outputs 'Yes' if there exists a proper k-colouring of the graph, is also easy to solve, even though the problem GRAPH-COLOURING defined above, whose input is a graph in 'adjacency matrix form' together with an integer $K \geq 1$, is widely believed not to have any quick algorithm.

Solving problems

We say a given machine M *decides* a language $\mathcal{L}$ if M halts on any input $x \in \Sigma^*$, answering 'Yes' if $x \in \mathcal{L}$ and 'No' if $x \notin \mathcal{L}$. In the next lecture, we'll also be interested in *accepting* $\mathcal{L}$, which is the weaker condition that M halts with 'Yes' if and only if $x \in \mathcal{L}$ (it may either answer 'No' or fail to halt if $x \notin \mathcal{L}$). If $\mathcal{L}$ is the language we get from a given decision problem (with our choice of input encoding) then we sometimes just say that M solves the problem. Observe that, by definition, if M decides a language at all, then it decides just one language.

Traditionally, Turing Machines that are supposed to solve problems have states called 'Yes' and 'No'; halting in one of these two states is supposed to indicate the machine's answer (but you could equally well do it by making the machine halt with the appropriate message on the tape).

Decidability

It's easy to see that there are languages which are not decided by any TM: there are a countably infinite number of possible TMs (for each number of states, there are only finitely many possibilities, and a countable union of finite sets is countable). But there are uncountably many languages—the power set of Σ^*, which is a countably infinite set, is the set of all languages. But are there any 'interesting' such languages? In particular, there are only countably many languages specified by a decision problem (for the same reason as TMs)—is any decision problem undecidable?

Theorem 14 (Turing, 1936)**.** *Fix a way of encoding the pair (M, I) where M is a TM and I an input. Then the language $\mathcal{L}_{Halting}$, consisting of all pairs (M, I) where M halts eventually on input I, is undecidable.*

Proof. Suppose to the contrary there is some TM H which decides $\mathcal{L}_{\text{Halting}}$. We construct a TM P which does the following:

(*i*) For input x, check if x is a string representing a TM M. Halt if not.

(*ii*) Simulate H with input (M, x).

(*iii*) If H returns 'Yes' go into an infinite loop, if H returns 'No' halt.

To justify that this is all possible, observe that (if you were given H) it would be easy to program the rest on a tape-computer (It's not all that hard to do 'by hand', in fact—try it as an exercise).

Now let p be the string representing the TM P. The result of running P with input p is a contradiction. $\square$

The Halting Problem is a fairly natural problem to want to solve—at least this is better than knowing undecidable languages exist by counting—but one can do better: Matiyasevich, in his 1970 PhD thesis, building on previous work of Davis, Putnam and Robinson, showed that the problem of whether a given Diophantine equation (a polynomial in many variables with integer coefficients) has integer solutions is undecidable. This problem was and is a central concern of number theory. Note that this does not 'show humans are better than computers'; a computer can simply check by brute-force all finite strings of symbols to see whether they constitute a valid proof, so anything a human can prove about (say) Diophantine equations a computer can in principle also prove.

In fact, the Church-Turing Thesis is the (non-mathematical!) assertion that any physically possible form of computation can be simulated by Turing Machines; there is no known counterexample (quantum computing, if it turns out to be physically possible, may give but is not known to give a counterexample to a much stronger statement which asserts that TMs are also at worst polynomially slower than any physically possible form of computing; TMs can simulate quantum computers, though they run exponentially slower using the best known algorithms).

Exercises

Exercise 1. *How does Euler's Formula for graphs embedded in the plane need to be modified to handle graphs with c components, where c is not necessarily equal to 1?*

Exercise 2. *Consider the oriented surface S_k, $k \geq 1$. Draw a graph in S_k by putting a vertex at each corner of the boundary $4k$-gon, and an edge along each segment of the boundary, identifying any of these edges and vertices as necessary. Count the number of vertices, edges, and faces in this embedding, and verify the Euler-Poincaré Formula in this case.*

Exercise 3. *(a) Describe the graphs not containing K_3 as a minor.*

(b) Describe the graphs not containing the 4-cycle C_4 as a minor.

Exercise 4. *Let P be the* Petersen graph. *(If you don't know what this is, find out!)*

(a) Show that P is non-planar in the following three ways:

- *using Euler's Formula;*
- *by showing that P contains $K_{3,3}$ as a topological minor;*
- *by showing that P contains K_5 as a minor.*

(b) What is the minimum size of a set F of edges of P whose deletion leaves a planar graph?

Exercise 5. *Show that the property of being bipartite is not closed for the topological minor ordering on graphs. (Very easy)*

Exercise 6. *Prove that the property of having no connected component with more edges than vertices is minor-closed.*

Find as many minimal forbidden minors as you can for this property.

Exercise 7. *Prove that the set of finite simple graphs $\mathscr{G}$ with the topological minor ordering $\leq_T$ is not a well-quasi-ordering. In other words, give an infinite sequence of graphs $G_1, G_2, \ldots$, for which there are no two indices i, j with $i < j$ and $G_i \leq_T G_j$.*

(This is a hard question. Feel free to do an Internet search, but you must show that the sequence you give has the desired property.)