Trees

On to something completely different: trees.

Let me first recall a few things, or, rather, fill a few gaps.

Def. If (V, E) is a simple graph, then we identify this simple graph (V, E) with the multigraph (V, E, id). Conversely, for each multigraph (V, E, φ), we can form the simple graph $(V, \varphi(E))$.

These are not exactly inverse constructions: The latter undoes the former, but the former does not exactly undo the latter (the "names" of the edges of the multigraph are lost, as is the information how many edges there are between 2 vertices).

Def. Let ~~(V, E, φ) be a circuit in a~~ $w = (v_0, e_1, v_1, e_2, v_2, \ldots, e_k, v_k)$ be a walk in a multigraph $G = (V, E, \varphi)$. We say that w is <u>backtrack-free</u> if

$$e_i \neq e_{i-1} \quad \forall i \in \{2, 3, \ldots, k\}.$$

~~We say that W is cyclically backtrack-free if W is backtrack-free and $e_1 \neq e_k$.~~

Prop. 11. ~~Let u and v be two ver~~ Any nonempty backtrack-free circuit in a multigraph contains a cycle. ~~(as a~~

("Nonempty" means length >0.

"Contains" means that a cycle can be obtained from it by dropping edges.)

Proof. Our circuit clearly contains two ~~appear~~ equal vertices. Choose a pair of two equal vertices as closely positioned on the circuit as possible ~~(both~~ (i.e., the shortest ~~closed~~ nonempty sub-circuit).

The stuff between them is a cycle.

Prop. 12. Let $G = (V, E, \varphi)$ be a multigraph. Let $u, v \in V$.

Let $\mathbf{a}$ and $\mathbf{b}$ be two distinct backtrack-free walks from u to v. Then, $\exists$ cycle in G.

Proof. ~~Write $\mathbf{a}$ as $(p_0, a_0, p_1, a_1,$~~

Write $\mathbf{a}$ as $(p_0, a_1, p_1, a_2, p_2, \ldots, a_k, p_k)$.

Write $\mathbf{b}$ as $(q_0, b_1, q_1, b_2, q_2, \ldots, b_\ell, q_\ell)$.

Let $p_i = q_i$ be the first vertex where the two walks diverge (i.e. we have $a_{i+1} \neq b_{i+1}$, provided that $i < k$ & $j < \ell$). Then,

$$(p_k, a_k, p_{k-1}, a_{k-1}, \ldots, p_{i+1}, a_{i+1},$$
$$p_i = q_i, b_{i+1}, q_{i+1}, b_{i+2}, q_{i+2}, \ldots, b_\ell, q_\ell)$$

is a backtrack-free nonempty circuit (since $p_k = v = q_\ell$ and $a_{i+1} \neq b_{i+1}$), and thus (by ~~Prop.~~ Prop. 11) contains a cycle. $\square$

Def. A <u>forest</u> is a multigraph with no cycles. (In particular, it thus cannot have 2 edges between the same 2 vertices. Hence, we can treat it as a simple graph if we don't care for the names of the edges.)

A <u>tree</u> is a ~~tr~~ connected forest.

Thm. 13 (the tree equivalence theorem).

Let $G = \cancel{}\ (V, E, \varphi)$ be a multigraph.

TFAE (this means "The Following Are Equivalent"):

<u>Statement T_1:</u> G is a tree.

<u>Statement T_2:</u> $V \neq \emptyset$, and for each $u \in V$ and $v \in V$, there is a <u>unique</u> path $u \to v$.

<u>Statement T_3:</u> $V \neq \emptyset$, and for each $u \in V$ and $v \in V$, there is a <u>unique</u> backtrack-free walk $u \to v$.

Statement T_4: G is connected, and $|E| = |V| - 1$.

Statement T_5: G is connected, and $|E| < |V|$.

Statement T_6: G is a forest, but adding any new edge to G produces a cycle.

Statement T_7: G is connected, but removing any edge from G yields a non-connected (we often say "disconnected") multigraph.

Statement T_8: EITHER $\exists\, v \in V$ such that $\deg v = 1$ and the induced subgraph $G|_{V \setminus \{v\}} = (V \setminus \{v\},$

$E' := \{e \in E \mid v \notin \varphi(e)\}, \varphi|_{E'})$

(this is simply what remains if we remove v and the unique edge through v) is ~~either~~ a tree, ?

~~has 0 vertices~~ OR $|V| = 1$.

<u>Statement τ_g:</u> G is a forest and

$|E| \geq |V| - 1$ and $V \neq \emptyset$.

<u>Rmk.</u> The notions of a "tree" used in computer science are often different from ours!

In particular, our trees have no pre-determined root (all vertices are "equal in rights"), and there is no concept of children vs. parents.

Before we prove Thm. 13, let us prepare some lemmas!

Def. Let $b_0(G)$ denote the # of connected components of a ~~graph~~ multigraph (or simple graph) G.

Prop. 14. Let $G = (V, E, \varphi)$ be a multigraph. Then,

$$b_0(G) \geq |V| - |E|.$$

Proof. Induction over $|E|$:

- Induction base: For $|E| = 0$, each vertex forms its own connected component $\Rightarrow b_0(G) = |V|$.
- Induction step: Adding 1 more edge can reduce $b_0(G)$ by 1 at most (since the new edge might connect two connected components, but never three or more). $\square$

Prop. 15. Let G be a multigraph, Let c be a cycle of G, Let e be an edge of c. Let $G \setminus e$ be G with the edge e removed. Then, $b_0(G \setminus e) = b_0(G)$.

Proof. Removing e from G does not disconnect any two vertices, because any walk that uses e can be re-routed through the rest of the cycle c.
Thus, the connected components of $G \setminus e$ (as sets of vertices) are exactly the connected components of G. $\square$

Cor. 16. Let $G = (V, E, \varphi)$ be a multigraph that is not a forest. Then,

$$b_0(G) \geq |V| - |E| + 1.$$

Proof. $\exists$ cycle c (since G is not a forest).

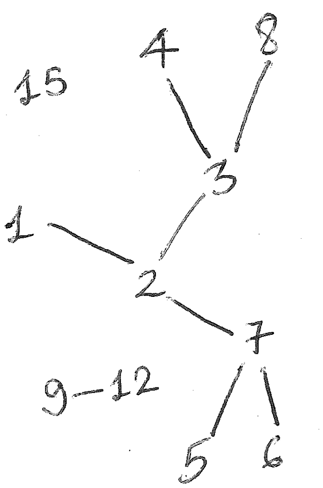
$\exists$ edge e in c. Now, apply Prop. 14 to $G \setminus e$. This yields

$$b_0(G \setminus e) \geq |V| - |E \setminus \{e\}| = |V| - |E| + 1.$$

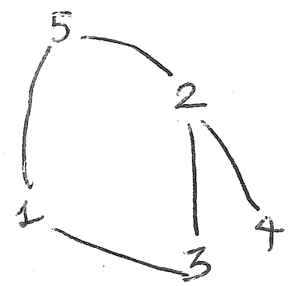
Rewrite using Prop. 15. $\square$

Prop. 17. Let G be a forest with ~~≥ 2 vertices. Then,~~ ≥ 1 edge.
Then, $\exists \geq 2$ vertices v satisfying $\deg v = 1$.

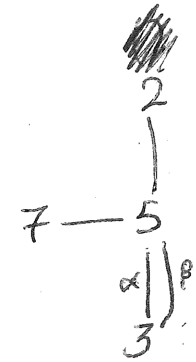
Such vertices are called <u>leaves</u>.



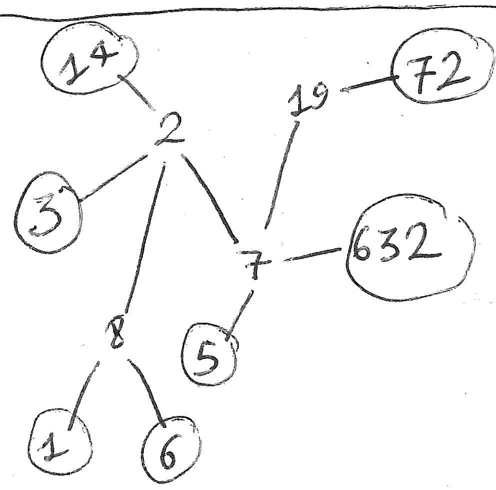
a forest (but not a tree)



not a forest



not a forest ((3, α, 5, β, 3) is a cycle!)



a tree (the leaves are circled)

<u>Proof.</u> Let $(v_0, a_1, v_1, a_2, v_2, \ldots, v_m)$ be a longest path in G. (Such a path clearly exists, since paths cannot

have length $\geq |V(G)|$.

I claim that v_0 is a leaf.
Indeed, assume the contrary. Thus, $\exists$ an edge a_0 through v_0 distinct from a_1. Let v_{-1} be its other end.

$\Rightarrow (v_{-1}, a_0, v_0, a_1, v_1, \ldots, v_m)$ is a backtrack-free walk.

This cannot be a path, since it would then be longer than the longest path. Compare it with an actual path $v_{-1} \to v_m$. Thus, by Prop. 12,

$\exists$ cycle in G. This contradicts G being a forest.

So we have shown that v_0 is a leaf. Similarly, v_m is a leaf.
Also, $m \neq 0$ (since $\exists$ edge $\Rightarrow$ longest path has length ≥ 1), thus $v_m \neq v_0$.
Hence we have found 2 leaves. $\square$

Cor. 18. Let $G = (V, E, \varphi)$ be a tree with $|V| \geq 2$. Let v be a leaf of G. Let $G \setminus \{v\}$ be the multigraph obtained from G by removing v and all edges through v. ~~Then~~ (Explicitly,

$$G\setminus\{v\} = (V\setminus\{v\},$$
$$E' := \{e \in E \mid v \notin \varphi(e)\},$$
$$\varphi|_{E'}).\ )$$

Then, $G\setminus\{v\}$ is a tree.

<u>Proof.</u> Clearly, $G\setminus\{v\}$ has no cycles (since G has none).
$\Rightarrow$ $G\setminus\{v\}$ is a forest.

Remains to show that $G\setminus\{v\}$ is connected.

Let p and q be two vertices in $G\setminus\{v\}$. Then, $\exists$ path $p \to q$ in G. This path must not contain v (since $\deg v = 1$, so we cannot enter and leave v without using the same edge twice in a row). $\Rightarrow$ It is a path in $G\setminus\{v\}$.

Hence, $\exists$ path $p \to q$ in $G\setminus\{v\}$. Thus, $G\setminus\{v\}$ is connected. $\square$

Cor. 18 allows us to prove facts about trees by induction!

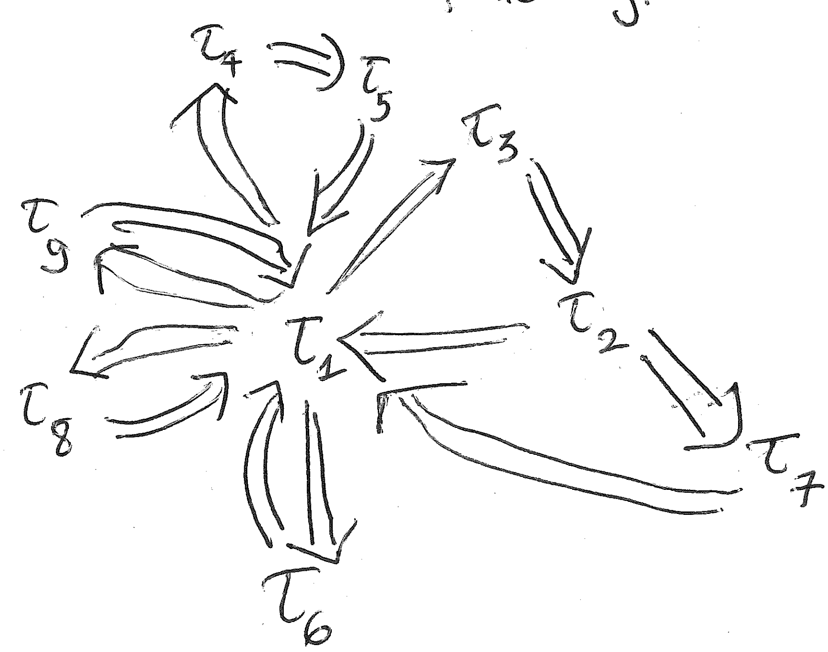
Prop. 19. Let G be a tree. Then, $|E(G)| = |V(G)| - 1$.

Proof. Induction over $|V(G)|$:

If $|V(G)| \geq 2$, then pick any leaf v (these exist by Prop. 17), and apply the induction hypothesis to $G \setminus \{v\}$. Then, observe that G has precisely 1 more vertex and 1 more edge than $G \setminus \{v\}$ (since v is a leaf). $\square$

Now, finally:

Proof of Thm. 13. We shall prove the following:

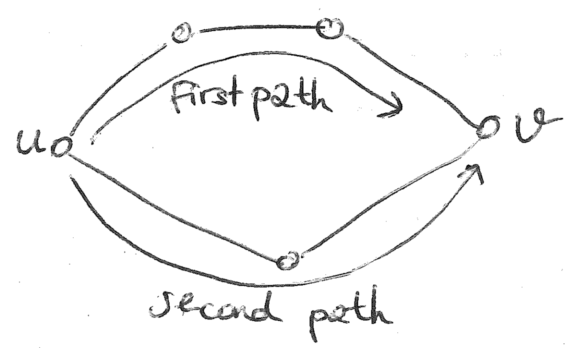


This digraph being strongly connected (i.e., you can travel from T_i to T_j $\forall i,j$), this will prove Thm. 13.

$T_1 \Rightarrow T_3$: If we had two distinct backtrack-free walks $u \to v$, then we would have a cycle (by Prop. 12), hence G would not be a tree.

$T_3 \Rightarrow T_2$: A path is a backtrack-free walk. And if $\exists$ walk $u \to v$, then $\exists$ path $u \to v$.

$T_2 \Rightarrow T_1$: If $\exists$ cycle, then $\exists$ two different paths between any two distinct vertices on this ~~path~~ cycle:



Hence, $\nexists$ cycle. Also, G is connected $\Rightarrow$ G is a tree.

$\underline{T_1 \Rightarrow T_6}$: Need to prove that adding any new edge uv produces a cycle.

Indeed, G is connected, so $\exists$ path $u \to v$. If we combine this path with the new edge uv, we get a cycle.

$\underline{T_6 \Rightarrow T_1}$: Need to show that G is connected.

Indeed, assume the contrary. Then, $\exists$ two vertices u and v in distinct connected components. Hence, adding a new edge uv must not produce a new cycle (since that cycle would have to cross over from the connected component of u into that of v ~~at least twice~~ and back again, but there is only one edge connecting them). Hence, T_6 is contradicted.

$\underline{T_2 \Rightarrow T_7}$: Connectedness is clear (since $\exists$ path). Now, let us remove an edge uv from G. Then, this edge was the only path $u \to v$ to G (since T_2 shows that the path $u \to v$ is unique), and so there is no path $u \to v$

anymore after we removed it.
$\Rightarrow$ We obtained a non-connected graph.

$\underline{T_7 \Rightarrow T_1:}$ Need to show G is a forest.

Assume the contrary. Then, $\exists$ cycle c. ~~Let~~ Fix any edge e on c.
Prop. 15 yields $b_0(G/e) = b_0(G) = 1$, so G/e is connected, contradicting T_6.

$\underline{T_1 \Rightarrow T_8:}$ Prop. 17 yields $\exists v \in V$ such that $\deg v = 1$.
Cor. 18 does the rest.
(This is for the case when $|V| \geq 2$. Else

$\underline{T_8 \Rightarrow T_1:}$ Need to show G has no cycles.

But any cycle of G is a cycle of $G|_{V \setminus \{v\}}$ (since v ~~...~~ is a dead end

and thus useless for cycles), and $G|_{V \setminus \{v\}}$ has no cycles.

$\underline{T_1 \Rightarrow T_4:}$ Just use Prop. 19.

$\underline{T_4 \Rightarrow T_5:}$ Clear.

$\underline{T_5 \Rightarrow T_1:}$ If G was not a forest, then Cor. 16 would yield

$$b_0(G) \geq |V| - |E| + 1 > 1$$
(since $|E| < |V|$),

so G would not be connected. But this would contradict T_5.
So G is a forest $\Rightarrow$ a tree.

$\underline{T_1 \Rightarrow T_9:}$ Prop. 19 again.

$\underline{T_9 \Rightarrow T_1:}$ Since G is a forest, each connected component of G is a tree, and thus has 1 fewer edge than it has vertices (by Prop. 19). So, altogether, G has $b_0(G)$ fewer edges than it has vertices (since each edge and each vertex belongs to exactly one connected component).

$\Rightarrow |E| = |V| - b_0(G)$.

$\Rightarrow |V| - b_0(G) = |E| \geq |V| - 1$

$\Rightarrow b_0(G) \leq 1 \Rightarrow G$ is connected (since $V \neq \emptyset$) $\Rightarrow G$ is a tree. $\square$

<u>Cor. 20.</u> If G is a forest, then

$$|E(G)| = |V(G)| - b_0(G).$$

<u>Proof.</u> See proof of Thm. 13, $t_g \Rightarrow T_1$. $\square$

Spanning trees

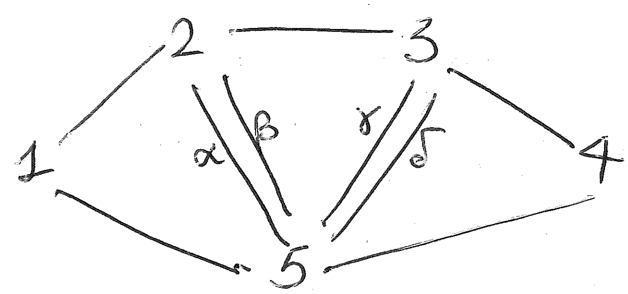
Def. A spanning subgraph of a multigraph $G = (V, E, \varphi)$ is a multigraph of the form $(V, F, \varphi|_F)$ with $F \subseteq E$.

(In other words, it is a submultigraph with the same vertex set as G.)

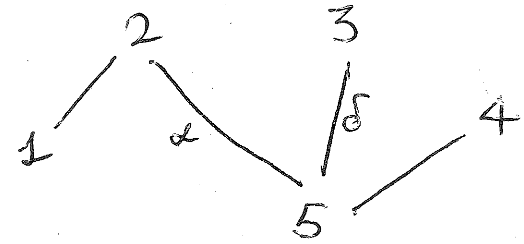
[Generally, a subgraph of a multigraph $G = (V, E, \varphi)$ means a multigraph of the form $(W, F, \varphi|_F)$ with $W \subseteq V$ and $F \subseteq E$ satisfying $\varphi(F) \subseteq \mathcal{P}_2(W)$.]

A spanning tree of a multigraph G means a ~~sta~~ spanning subgraph of G that is a tree.

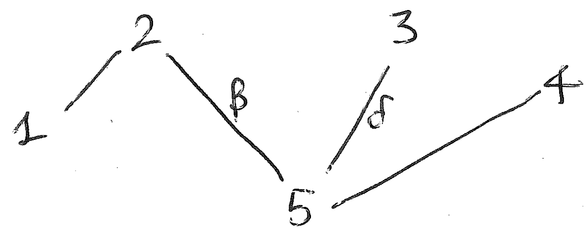
Example: Here is a multigraph G:



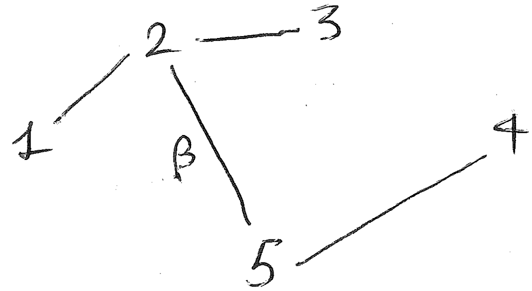
Here is a spanning tree:



And here is another ($\alpha \neq \beta$!):



And another:



<u>Rmk:</u> A <u>spanning forest</u> of a multigraph G means a spanning subgraph H of G that is a forest and satisfies $b_0(H) = b_0(G)$,

When G is connected, this is the same as a spanning tree of G.

<u>Thm. 21.</u> Each connected multigraph G has at least one spanning tree.

<u>First proof.</u> Keep removing edges from G in such a way that G remains connected.
At some point, this becomes no longer possible. Thus, you found a ~~sub~~ spanning subgraph H of G such that H is connected, but removing any edge from G yields a non-connected multigraph.

By Thm. 13 ($T_7 \Rightarrow T_1$, applied to H instead of G), this yields that H is a tree. Thus, H is a spanning tree of G.

<u>Second proof.</u> We recursively construct a subgraph of G that is a tree (but not necessarily spanning):

First, choose any vertex r of G.

Recursion base: Set $V_0 := \{r\}$ and $E_0 := \emptyset$.

Recursion step: Let $i \in \mathbb{N}$. Assume that $V_0, \ldots, V_i$ and $E_0, \ldots, E_i$ are already defined. Want to define V_{i+1} (a set of vertices of G) and E_{i+1} (a set of edges of G).

Set $V_{i+1} :=$ {vertices of G that have a neighbor in V_i but don't belong to $V_0 \cup V_1 \cup \cdots \cup V_i$}.

For each $v \in V_{i+1}$, choose <u>one</u> edge e_v connecting v to a neighbor in V_i. (Such an edge exists, since $v \in V_{i+1}$. If there are many, choose one as you like.)

Set $E_{i+1} := \{e_v \mid v \in V_{i+1}\}$.

At some i, you will find that $V_{i+1} = \emptyset$. Stop the recursion there. Then you have

$$V_0 \cup V_1 \cup \cdots \cup V_i = V(G).$$

Moreover, $\forall\, k \in \mathbb{N}$,

$V_k = \{ v \in V(G) \mid$ ~~the~~ a shortest path $r \to v$ has length $k \}$.

(This is easily proven by induction.)

Now, $(V,\ E_1 \cup E_2 \cup \dots \cup E_k,\ \varphi|_{E_1 \cup E_2 \cup \dots \cup E_k})$

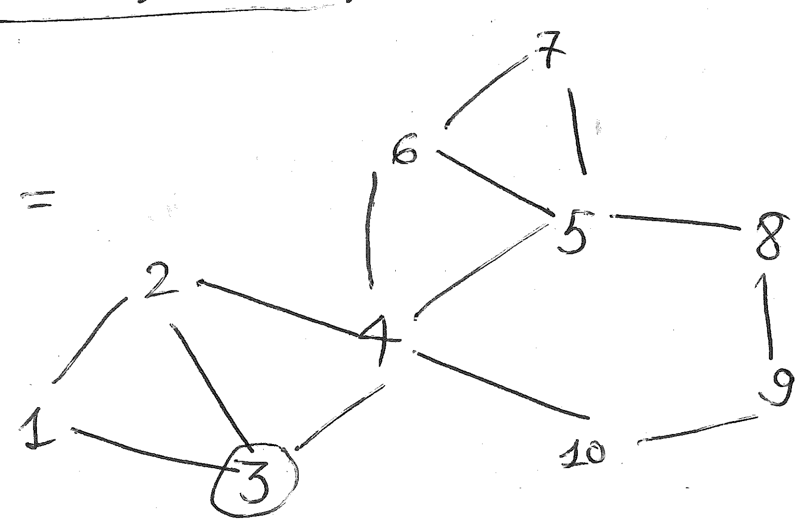
is a spanning tree of G.

It has the following extra property:

$\forall v \in V(G)$, the path $r \to v$ in this tree is a shortest path $r \to v$ in G. It is called a <u>breadth-first search ("BFS") tree</u>.

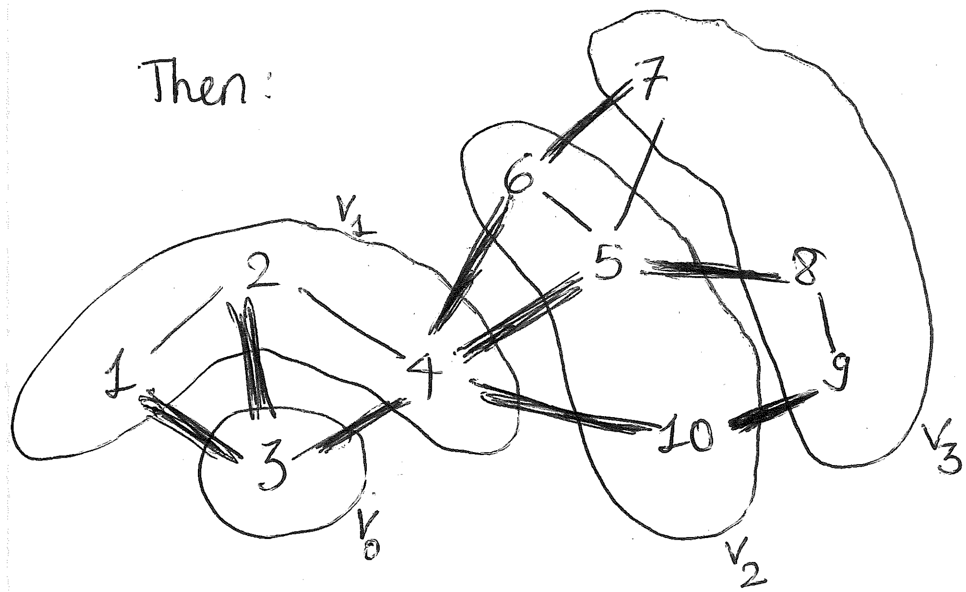
<u>Example:</u>

Let G =



Set $r = 3$.

Then:



where the bold edges are <u>one</u> possible choice you can make for $E_0 \cup E_1 \cup \dots \cup E_k$.

As you can see, they form a spanning tree.

Actually, this is simply the Dijkstra shortest-path algorithm, repurposed for the construction of a spanning tree.

<u>Third proof</u>: Another way is to ~~start~~ "explore the graph G" by following edges, each time taking an edge to a neighbor that has not already been explored. If such

an edge does not exist, then backtrack to the previous vertex and look for another edge. If such an edge doesn't exist there either, backtrack again, etc.

Once you have returned to the original vertex (where you started), you have explored all vertices.

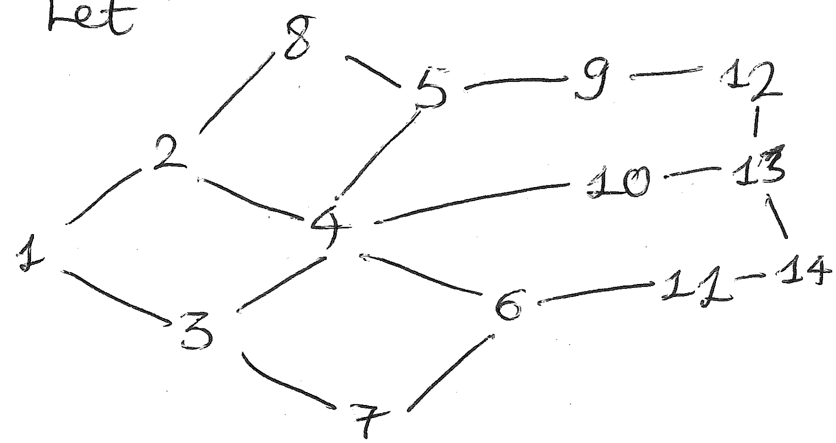
The edges you have traversed form a spanning tee, called a <u>depth-first search ("DFS") tree</u>.

It has the following extra property:

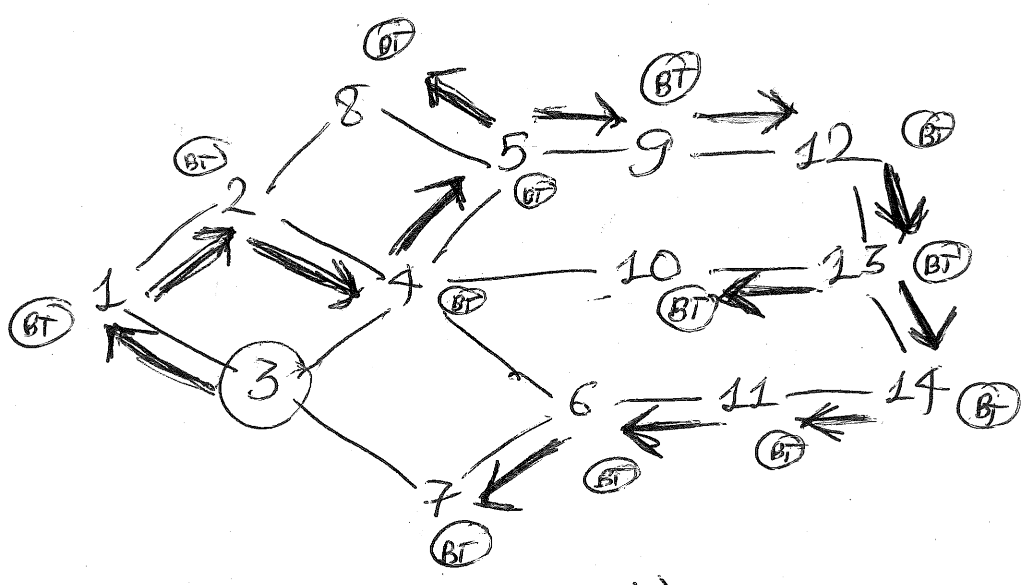
~~$\forall (u,v) \in E(G)$~~ If two vertices u and v of G are adjacent, then either u lies on the path from r to v in the tree, or v lies on the path from ~~v to~~ r to u in the tree. Here, r denotes the vertex on which you started exploring. (This is called a ~~For details~~ "lineal spanning tree".)

~~See~~ See [BenWil12, §6.1] for details.

Example: Let

$G =$ 

and pick $r = 3$. Then one possible ~~traver~~ exploration is:



(where (BT) = backtrack).

Observe that the result of the exploration is a spanning tree.

Cor. 22. Each ~~g~~ multigraph has a spanning forest.

Proof. Apply Thm. 21 to each connected component. Then, combine the resulting spanning trees into a spanning forest.