1 Matchings with Tutte's Theorem

Last week we saw a fairly strong necessary criterion for a graph to have a perfect matching. Today we see that this condition is in fact sufficient.

Theorem 1 (Tutte, '47)**.** *If there is no set $S \subseteq V(S)$ such that $G-S$ has more than $|S|$ components with an odd number of vertices, then G has a perfect matching.*

Proof. Let us refer to the condition given above that "there is no set $S \subseteq V(S)$ such that $G - S$ has more than $|S|$ components with an odd number of vertices", for brevity, as Tutte's criterion. We already showed previously that any graph not satisfying Tutte's criterion cannot have a perfect matching; we thus now want to show that all graphs satisfying Tutte's criterion do have a perfect matching.

One method we can use to further specify the set of graphs over which we have to prove this is to observe the effect of adding an edge to a graph which meets Tutte's Criterion results in a graph which still meets Tutte's Criterion. Suppose G satisfies Tutte's criterion; it is fairly easy to show that $G' = G + e$ satisfies it as well. Consider a set S of vertices from the shared vertex-set of G and G'. We know that $G - S$ has no more than $|S|$ odd components. $G' - S$ is either identical to $G - S$ (if e is incident on a vertex of S) or identical to $G - S + e$ (otherwise). In the first instance, $G' - S$ clearly has the same number of odd components as $G - S$. In the second, either e is internal to a component of $G - S$, in which case $G' - S$ has the same number of odd components as $G - S$, or it is between two components of $G - S$, in which case the number of components in $G' - S$ may be different, but it may be easily seen that $G' - S$ has no more odd components than $G - S$: an edge bridging two even components, or connecting an odd component to an even component, has no effect on the number of odd components, while an edge connecting two odd components (and thereby joining them into an even component) reduces the number of odd components by two. Thus, the number of odd components in $G' - S$ is at most equal to the number of odd components in G, so if G satisfies the Tutte criterion, so does G'. It is even easier, of course, to show that if G has a perfect matching, so does G'. As a practical upshot of this, we can see that if G satisfies both the condition and consequence of Tutte's theorem, so does any graph resulting from adding edges to G, so we only need demonstrate Tutte's theorem for *edge-minimal* graphs which satisfy Tutte's criterion.

We may thus inspect a very specific set of graphs. Starting from a graph consisting only of an even number of isolated vertices (if $|G|$ is odd, it clearly does not satisfy Tutte's criterion, and is not of interest here) and adding edges sequentially, we are guaranteed two phases, and we posit the possibility of a third: when there are very few edges, both Tutte's criterion and existence of a perfect matching are false. This sequence terminates in a complete graph K_{2n}, which has both a perfect matching and Tutte's criterion, so at some point Tutte's criterion is achieved, and once achieved for some graph, we saw above that it is also achieved for every graph further along in the sequence; likewise once a graph in the sequence has a perfect matching, so does every subsequent graph. Thus, our interest is in ascertaining that perfect matching and the Tutte criterion are achieved simultaneously in the sequence. We posit the existence of a *maximal counterexample* to Tutte's Theorem: that is, a graph G such that G satisfies Tutte's criterion, and G does not have a perfect matching, but that $G+e$ has a perfect matching for any edge e not appearing in G. We shall show that such a graph actually must have a perfect matching, contradicting our presumption.

Let us consider S consisting of all vertices in G of degree $|G|-1$ (note that this is the set of vertices which are adjacent to every other vertex in G, and that S may well be empty). We shall now

inspect two cases which may arise in examining the structure of $G - S$:

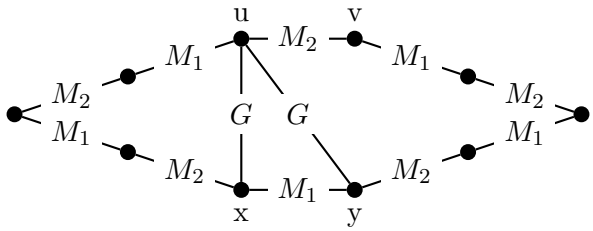
Case I: every component of $G - S$ is a complete graph. Then, by the Tutte criterion, $G - S$ has at most $|S|$ components which are cliques of odd size, and all the other components are cliques of even size. Within a clique of even size, finding a perfect matching is trivial; within a clique of odd size, finding a matching which covers every vertex but one is likewise trivial. Thus, if we have k odd components, it is easy to find a matching on all but $|S| + k$ vertices; furthermore, since every element of S is adjacent to every other vertex in the graph, and since $|S| > k$, we can match the k leftover vertices with vertices of S, leaving only $|S| - k$ vertices in S unmatched. These vertices are mutually adjacent, so if there are an even number of them, matching them is easy. Since $|G|$ must itself be even to satisfy Tutte's criterion, the number of leftover vertices after a partial matching must be even. Thus, in this case, we have an extremely straightforward construction of a perfect matching.

Case II: some component of $G - S$ is not a complete graph. This component thus has nonadjacent vertices, which are at a distance of at least 2; we can specifically choose vertices x and y at a distance of exactly 2, so that x and y are nonadjacent but have a mutual neighbor u. In addition, since $u \notin S$, there is at least one vertex to which u is not adjacent; let us call it v. Our maximal-counterexample condition guarantees that adding the edge $\{x, y\}$ or $\{u, v\}$ to G creates a graph with a perfect matching; let $M_1 \subseteq [E(G) \cup \{x, y\}]$ and $M_2 \subseteq [E(G) \cup \{u, v\}]$ be the aforementioned perfect matchings on larger graphs. Clearly $\{x, y\} \in M_1$ and $\{u, v\} \in M_2$, since otherwise M_1 or M_2 would be a perfect matching on G itself. Let F consist of all edges which appear in either M_1 or M_2, but not in both. Note that $\{x, y\}$ and $\{u, v\}$ will both be in F.

Since M_1 and M_2 are perfect matchings, every vertex of G is incident on exactly one edge in M_1 and exactly one edge in M_2. If these edges are identical, then they definitionally do not appear in F; if they are non-identical, then they both appear in F. Thus, every vertex in G is incident on either 0 or 2 edges in F. If we consider a graph H with $V(H) = V(G)$ and $V(H) = F$, then H thus consists of isolated vertices (of degree zero) and cycles (consisting of vertices of degree 2). Furthermore, we know there are an even number of isolated vertices, which can be paired off according to the edge (not appearing in F) which is shared between M_1 and M_2; likewise, since each vertex in a cycle is incident on one edge from M_1 and one edge from M_2, we know these cycles are of even order, with edges alternatingly from M_1 and M_2.

Since $\{x, y\} \in F$, there is a cycle in H containing x and y. If u and v are not in this cycle, then we may construct a matching on G by taking the edges of M_2 which are in this cycle and the edges of M_1 which are not in this cycle: this is a matching much like M_1, but shifted within this cycle to avoid the edge $\{x, y\}$, so it is a matching on G.

Alternatively, if x, y, u, and v are all in the same cycle of H, we have a very good idea of its structure. The edge $\{x, y\}$ is from M_1, and there are an odd number of intervening vertices before u and v appear, in some order. x and y are interchangable, so we may arbitrarily assume an orientation resembling that shown below:



Note that the edges labeled with "G" above appear in G but not in H, while the edges $\{u, v\}$ and $\{x, y\}$ not appear in H but not in G; all other edges pictured appear in both G and H. Note that the edge $\{u, y\}$ divides this structure up into two even cycles sharing the edge $\{u, y\}$ so a matching on this component can be constructed using only edges from G: the edge $\{u, y\}$ and edges from M_1 to its left, and edges from M_2 to its right. On every other component of H, we may use edges from either M_1 or M_2 to build the remainder of our mathing, thus getting a perfect matching on G. □

One can use this theorem to prove several classes of graph to have perfect matchings. One of the more popular characterizations is due to Petersen.

Definition 1. A *bridge* in a graph G is an edge e such that $G - e$ has more components than G.

Corollary 1 (Petersen, 1891)**.** *A bridgeless 3-regular graph has a perfect matching.*

Proof. We shall show that G satisfies Tutte's criterion for an arbitrary $S \subseteq V(G)$. If $G - S$ has no odd components, then it is obvious that the number of odd components of $G - S$ is less than $|S|$. Otherwise, let H be an odd component of $G - S$. Then, let m be the number of edges in G between H and S. We know that every vertex of H has degree 3 in G, so $\sum_{v \in H} d_G(v) = 3|H|$; since each edge between H and S induces a decrease in the degree of a vertex in H by 1, we know that $\sum_{v \in H} d_H(v) = 3|H| - m$. This is necessarily equal to $2\|H\|$, so it must be even. $|H|$ was defined to be odd, so m must also be odd in order to have an even difference. If m were 1, then this edge would necessarily be a bridge, since its removal disconnects S from H. Thus, $m \geq 3$.

We thus know that every odd component of $G - S$ is incident on at least three edges whose other endpoint is in S. Thus, if there are k odd components, then there are at least $3k$ edges between S and $G - S$. We thus know that $\sum_{v \in S} d_G(v) \geq 3k$. Since $d_G(v) = 3$ for all vertices, it follows that $3|S| \geq 3k$, so $|S| \geq k$, satisfying Tutte's criterion. □

Note that this result predates Tutte's Theorem by half a century, and was clearly not originally proven by appeal to Tutte's criterion.

2 Connectivity, part 2

We defined connectivity some time back. Now we're going to discuss it in the context of fault-tolerance: a network may well be connected, but will it remain connected when nodes or edges are removed? This clearly has real-world applications to transportation and communications networks: how does an interstate highway system, an airline's flight plan, or the Internet handle such disasters as road closure (edge removal), airport closure (vertex removal), or failure of either a router or a length of cable (edge or vertex removal). It is to this end that we introduce the *connectivity* parameters, which measure how fault-tolerant a network is.

Definition 2. A *vertex-cutset* of a graph G is a set of vertices $V \subseteq V(G)$ such that $G - V$ is disconnected or an isolated vertex. An *edge-cutset* (or simply *cutset*) is a set of edges $E \subseteq E(G)$ such that $G - E$ is disconnected. A vertex v is called a *cutvertex* if $G - v$ is disconnected, and an edge e is a *bridge* or *cut-edge* if $G - e$ is disconnected.

So this is a qualification of fault-tolerance, anasmuch as it gives a name to failure-points, but it's not a quantification, which attaches a specific quantity to fault-tolerance. To that end we introduce the following concepts:

Definition 3. A graph G is *k-vertex-connected* (or simply *k-connected*) if $|G| > k$ and for any set of vertices $V \subseteq V(G)$ with $|V| < k$, $G - V$ is connected; alternatively, G is k-connected if every vertex-cutset of G has at least k elements.

Likewise, G is *k-edge-connected* if, for any set of edges $E \subseteq E(G)$ with $|E| < k$, $G - E$ is connected, or if every edge-cutset of G has at least k elements.

The least k such that G is k-connected is called the *connectivity* of G, denoted $\kappa(G)$. The least k such that G is k-edge-connected is called the *edge-connectivity* of G, denoted $\kappa'(G)$.

Note that the connectivity and edge-connectivity of a graph could also be interpreted as the cardinality of its smallest vertex-cutset and edge-cutset respectively.

There are several graphs for which we know connectivity properties out-of-hand, so we set down a few simple observations here:

- If G is not connected, $\kappa(G) = \kappa'(G) = 0$.
- "connectedness", "1-connectedness", and "1-edge-connectedness" are identical concepts, except on the graph of one vertex.
- For a tree T, $\kappa'(T) = \kappa(T) = 1$
- $\kappa(K_n) = n - 1$.
- $\kappa(C_n) = \kappa'(C_n) = 2$, regardless of n.

We'd like to be able to describe connectedness structurally: what does a 2-connected, or 3-connected, or suchlike graph look like? Note that, unlike connectedness or tree structure, we cannot build a 2-connected graph one vertex at a time: there is no 2-connected subgraph of C_n.

We focus our energies chiefly on vertex-connectivity, which lends itself to more interesting theoretical results. For instance we can develop one simple result:

Proposition 1. *For any graph G, $\kappa(G) \leq \delta(G)$ and $\kappa'(G) \leq \delta(G)$.*

Proof. Let u be a vertex of G of minimum degree, so $d_G(u) = \delta(G)$; specifically let us label the neighbors of u as $v_1, v_2, \ldots, v_{\delta(G)}$. Let $V = \{v_1, v_2, \ldots, v_{\delta(G)}\}$; then, u has no neighbors in $G - V$, so either u is an isolated vertex of a larger graph or $G - V$ consists of a single vertex; in either case, V would be a cutset, so $\kappa(G) \leq |V| = \delta(G)$.

Likewise, if $E = \{\{u, v_1\}, \{u, v_2\}, \ldots, \{u, v_{\delta(G)}\}\}$, then $G - E$ has u isolated, so E is an edge-cutset, so $\kappa'(G) \leq |E| = \delta(G)$. □

2.1 Block structure of a connected graph

A 2-connected graph is one without cut-vertices. We shall explore the structure of cutvertex-free subgraphs of an arbitrary graph G.

Definition 4. A *block* of a connected graph G is an induced subgraph B which has no cutvertex, and such that B is not a subgraph of a larger induced subgraph without a cutvertex.

What may not be intuitively obvious is that blocks can be used to partition the edges of a graph, and, to a certain extent, to partition the vertices.

Proposition 2. *If B_1 and B_2 are distinct blocks of a connected graph G, then B_1 and B_2 share at most one vertex.*

Proof. Note that by the maximality criterion it is clear that neither $B_1 \subset B_2$ or $B_2 \subset B_1$. Suppose $V(B_1) \cap V(B_2) \geq 2$: we shall see that the graph formed by joining B_1 and B_2 thus has no cutvertex.

Suppose $\{u_1, u_2\} \in V(B_1) \cap V(B_2)$; let B be the graph consisting of all vertices and edges from B_1 and B_2. We shall show that $B - v$ is connected regardless of what vertex v is.

Case I: $v \in B_1$ and $v \notin B_2$. Since B_1 is a block, $B_1 - v$ is connected, so there is a path between any two vertices in $B_1 - v$, which is also a path in $B - v$. Likewise, since B_2 is a block, B_2 is connected, so there is a path between any two vertices in B_2, which is also a path in $B - v$. Finally, if $v \in B_1$ and $w \in B_2$, then we may find a walk between these two in $B - v$ produced by taking the already-found walk from v to u_1 in B_1 and from u_1 to w in B_2 and grafting them together.

Case II: $v \in B_2$ and $v \notin B_1$. Identical to Case I, with reversal of roles of B_1 and B_2.

Case III: $v \in B_1 \cap B_2$. Without loss of generality, let $v = u_2$. The argument in Case I, which makes use of routing via u_1, will still work to show that $B - v$ is connected.

Since we have shown that two blocks overlapping in more than one vertex must lie in a larger 1-connected set, it thus follows from maximality of blocks that two blocks cannot overlap in more than one vertex. □

Corollary 2. *If blocks B_1 and B_2 intersect in a vertex u, u is a cutvertex of $B_1 \cup B_2$.*

Proof. With regard to connectivity after cuts, cases I and II in the above proof hold even if $|B_1 \cap B_2| = 1$. Thus, in order for $B_1 \cup B_2$ to not be a block, it must be the case that removing a vertex in $B_1 \cap B_2$ results in a disconnected graph. □

Corollary 3. *Every edge of a connected graph G lies in exactly one block.*

Proof. By the previous proposition, no two vertices lie in multiple blocks, so no edge can lie in multiple blocks either. An edge is, in itself, a graph without a cutvertex, so either an edge is a block or it is a subgraph of a block. □

One interesting upshot of blocks' limited intersection is that we can describe their structure graphically as well: if G is a connected graph, then the *block diagram* of G is a graph whose vertex-set consists of the blocks of G and the vertices where blocks intersect, and in which a block and vertex are adjacent if the vertex lies in the block. Looking at these block diagrams, we actually find that they have strictly determined structure of their own.

Proposition 3. *The block diagram $B(G)$ of a connected graph G is connected.*

Proof. Let B_1 and B_2 be blocks of G, and let v_1 and v_2 be vertices in B_1 and B_2 respectively. Since G is connected, there is a path from v_1 to v_2: we shall include its edges in describing the walk as such: $\{v_1 = u_0, e_1, u_1, e_2, u_2, e_3, \ldots, e_{k-1}, u_{k-1}, e_k, u_k = v_2\}$. Since the blocks partition the edges, let us take note of which block each edge is in, letting $e_i \in B_i'$. Note that definitionally, since

$e_i \in B'_i$, it follows that u_{i-1} and u_i are in B'_i. We shall thus have a translation of our path in G to a walk (actually a path, but we don't need such specificity) on $B(G)$, as such:

Every sequence of edges $e_i, e_{i+1}, e_{i+2}, \ldots, e_{i+\ell}$ which all lie in a single block correspond to the vertex B'_i. Whenever e_i and e_{i+1} lie in different blocks, we know that u_i lies in both of those blocks, so u_i is a vertex of $B(G)$, which is adjacent to both B'_i and B'_{i+1}.

Thus we can construct a walk:

$$B'_1 = B'_2 = \cdots = B'_{i_1} \sim v_{i_1} \sim B'_{i_1+1} = B'_{i_1+2} = \cdots = B'_{i_2} \sim v_{i_2} \sim B'_{i_2+1} = \cdots = B'_{i_3} \sim \cdots \sim B'_k$$

This is a walk in $B(G)$ from B'_1 to B'_k, which may not be a walk from B_1 to B_2. Since $v_0 \in B_1$ and $v_0 \in B'_1$, we know that either $B_1 = B'_1$ or $B'_1 \sim v_0 \sim B_1$, so either B_1 is our walk's terminus or our walk has a trivial extension to B_1. Likewise, either $B'_k = B_2$ or $B'_k \sim v_k \sim B_2$.

To find a walk between a cutvertex v and a block B, we start by indentifying a block in G such that $v \in B'$, then graft the step $v \sim B'$ onto our guaranteed walk from B' to B. Likewise, to find a walk from cutvertex u to cutvertex v, find blocks such that $u \in B$ and $v \in B'$, and graft the steps $u \sim B$ and $v \sim B'$ onto the walk from b to B'. □