Staff Solutions to In-Class Problems Week 7, Mon.

STAFF NOTE: Digraphs: Walks & Paths Ch10-10.4

Problem 1.
Suppose that there are $n$ chickens in a farmyard. Chickens are rather aggressive birds that tend to establish dominance in relationships by pecking; hence the term “pecking order.” In particular, for each pair of distinct chickens, either the first pecks the second or the second pecks the first, but not both. We say that chicken $u$ virtually pecks chicken $v$ if either:

- Chicken $u$ directly pecks chicken $v$, or
- Chicken $u$ pecks some other chicken $w$ who in turn pecks chicken $v$.

A chicken that virtually pecks every other chicken is called a king chicken.

We can model this situation with a chicken digraph whose vertices are chickens with an edge from chicken $u$ to chicken $v$ precisely when $u$ pecks $v$. In the graph in Figure 1, three of the four chickens are kings. Chicken $c$ is not a king in this example since it does not peck chicken $b$ and it does not peck any chicken that pecks chicken $b$. Chicken $a$ is a king since it pecks chicken $d$, who in turn pecks chickens $b$ and $c$.

In general, a tournament digraph is a digraph with exactly one edge between each pair of distinct vertices.

(a) Define a 10-chicken tournament graph with a king chicken that has outdegree 1.

Solution. 1 pecks 2 and 2 pecks 3–10 and 3–10 peck 1. The directions of edges amongst 3-10 are irrelevant.

(b) Describe a 5-chicken tournament graph in which every player is a king.

Solution. An example is illustrated in Figure 2.
(c) Prove Theorem (King Chicken Theorem). The chicken with the largest outdegree in an $n$-chicken tournament is a king.

**Solution.** Proof. By contradiction. Let $u$ be a node in a tournament graph $G = (V, E)$ with maximum outdegree and suppose that $u$ is not a king. Let $Y = \{v \mid \langle u \rightarrow v \rangle \in E \}$ be the set of chickens that chicken $u$ pecks. Then $\text{outdeg}(u) = |Y|$.

Since $u$ is not a king, there is a chicken $x \notin Y$ (that is, $x$ is not pecked by chicken $u$) and that is not pecked by any chicken in $Y$. Since for any pair of chickens, one pecks the other, this means that $x$ pecks $u$ as well as every chicken in $Y$. This means that

$$\text{outdeg}(x) = |Y| + 1 > \text{outdeg}(u).$$

But $u$ was assumed to be the node with the largest degree in the tournament, so we have a contradiction. Hence, $u$ must be a king.

The King Chicken Theorem means that if the player with the most victories is defeated by another player $x$, then at least he/she defeats some third player that defeats $x$. In this sense, the player with the most victories has some sort of bragging rights over every other player. Unfortunately, as Figure 1 illustrates, there can be many other players with such bragging rights, even some with fewer victories.

**Problem 2.**

A 3-bit string is a string made up of 3 characters, each a 0 or a 1. Suppose you’d like to write out, in one string, all eight of the 3-bit strings in any convenient order. For example, if you wrote out the 3-bit strings in the usual order starting with 000 001 010. . . , you could concatenate them together to get a length $3 \cdot 8 = 24$ string that started 000001010. . . .

But you can get a shorter string containing all eight 3-bit strings by starting with 00010. . . . Now 000 is present as bits 1 through 3, and 001 is present as bits 2 through 4, and 010 is present as bits 3 through 5, . . . .

(a) Say a string is 3-good if it contains every 3-bit string as 3 consecutive bits somewhere in it. Find a 3-good string of length 10, and explain why this is the minimum length for any string that is 3-good.

**Solution.** The string 0001110100 is a length 10 string that is 3-good. You can’t do better: there must be two bits to start and each additional bit can yield at most one new 3-bit string.

(b) Explain how any walk that includes every edge in the graph shown in Figure 3 determines a string that is 3-good. Find the walk in this graph that determines your 3-good string from part (a).
Solution. A string can be built up from any walk by starting with the \( k \) bits in the vertex at the start of the walk and successively adding the bit that labels the edge to the end of the string being built. If the walk includes every edge, then any string \( b_1 b_2 b_3 \) will appear as a substring when the edge \( (b_1 b_2 \rightarrow b_2 b_3) \) appears in the walk.

In particular, the string \( 00110100 \) is determined by the walk that goes through the following sequence of edges:

\[
\langle 00 \rightarrow 00 \rangle \langle 00 \rightarrow 01 \rangle \langle 01 \rightarrow 11 \rangle \langle 11 \rightarrow 11 \rangle \langle 11 \rightarrow 10 \rangle \langle 10 \rightarrow 01 \rangle \langle 01 \rightarrow 10 \rangle \langle 10 \rightarrow 00 \rangle.
\]

(c) Explain why a walk in the graph of Figure 3 that includes every edge exactly once provides a minimum-length 3-good string.\(^1\)

Solution. Since there are 8 edges, the string determined by the walk will be of length 10, which is the minimum possible as observed in part (a). Since the walk includes every edge, it will determine a 3-good string by part (b).

(d) Generalize the 2-bit graph to a \( k \)-bit digraph, \( B_k \), for \( k \geq 2 \), where \( V(B_k) := \{0,1\}^k \), and any walk through \( B_k \) that contains every edge exactly once determines a minimum length \( (k + 1) \)-good bit-string.\(^2\)

What is this minimum length?

Define the transitions of \( B_k \). Verify that the in-degree and out-degree of every vertex is even, and that there is a positive path from any vertex to any other vertex (including itself) of length at most \( k \).

Solution. \( 2^{k+1} + k \).

A bit-string of length \( n \) has exactly \( n - k \) locations where a length \( k + 1 \) subsequence can begin. Since there are \( 2^{k+1} \) length-\( k + 1 \) bit-strings, the minimum length, \( n \), of any \( (k + 1) \)-good bit-string must satisfy \( n - k \geq 2^{k+1} \), so the minimum length is \( 2^{k+1} + k \). This is exactly the length of the bit-string that would be determined by a walk containing all \( 2 \cdot 2^k \) edges, \( E(B_k) \), in the graph \( B_k \):

\[
E(B_k) := \{\langle ax \rightarrow xb \rangle \mid x \in \{0,1\}^{k-1} \text{ AND } a, b \in \{0,1\}\}
\]

If \( y \in \{0,1\}^k \), then \( y = ax \) and \( y = zb \) for unique strings \( x, z \in \{0,1\}^{k-1} \) and bits \( a, b \in \{0,1\} \). Then by definition of \( E(B_k) \), there are exactly two edges out of \( y \), one going to \( x0 \) and the other to \( x1 \), so \( \text{outdeg}(y) = 2 \). Likewise, there are exactly two edges into \( y \), one from \( 0z \) and the other from \( 1z \), so \( \text{indeg}(y) = 2 \).

To get from vertex \( b_1 b_2 \ldots b_k \) to \( c_1 c_2 \ldots c_k \) with a length \( k \) walk, proceed as follows:

\[
\begin{align*}
&b_1 b_2 b_3 \ldots b_k \rightarrow b_2 b_3 \ldots b_k c_1 \rightarrow b_3 \ldots b_k c_1 c_2 \\
&\quad \rightarrow \cdots \rightarrow b_k c_1 c_2 \ldots c_{k-1} \rightarrow c_1 c_2 \ldots c_k.
\end{align*}
\]

Since a walk of length \( k \) exists, a path of length at most \( k \) can be obtained by removing the cycles in the walk.\(^\blacksquare\)

\(^1\)The 3-good strings explained here generalize to \( n \)-good strings for \( n \geq 3 \). They were studied by the great Dutch mathematician/logician Nicolaas de Bruijn, and are known as \textit{de Bruijn sequences}. de Bruijn died in February, 2012 at the age of 94.

\(^2\)Problem 10.8 explains why such “Eulerian” paths exist.
Problem 3.

An Euler tour\(^3\) of a graph is a closed walk that includes every edge exactly once. Such walks are named after the famous 17th century mathematician Leonhard Euler. (Same Euler as for the constant \(e \approx 2.718\) and the totient function \(\phi\) — he did a lot of stuff.)

So how do you tell in general whether a graph has an Euler tour? At first glance this may seem like a daunting problem (the similar sounding problem of finding a cycle that touches every vertex exactly once is one of those million dollar NP-complete problems known as the Hamiltonian Cycle Problem)—but it turns out to be easy.

(a) Show that if a graph has an Euler tour, then the in-degree of each vertex equals its out-degree.

Solution. Let

\[ C := v_1 \langle v_1 \to v_2 \rangle v_2 \ldots \langle v_r \to v_1 \rangle v_1 \]

be an Euler tour. Except for the initial and final occurrences of \(v_1\), each occurrence of a vertex \(v\) in the tour is immediately preceded by an edge \(\langle u \to v \rangle\) and immediately followed by an edge \(\langle v \to w \rangle\). It follows that if \(v \neq v_1\) occurs \(s\) times in \(C\), then \(\text{indeg}(v) = \text{outdeg}(v) = s\) since every edge incident to \(v\) occurs in \(C\) exactly once. For the same reason, if \(v_1\) occurs \(s\) times on the path, then

\[ \text{indeg}(v_1) = \text{outdeg}(v_1) = s - 1. \]

A digraph is weakly connected if there is a “path” between any two vertices that may follow edges backwards or forwards.\(^4\) In the remaining parts, we’ll work out the converse. Suppose a graph is weakly connected, and the in-degree of every vertex equals its out-degree. We will show that the graph has an Euler tour.

A trail is a walk in which each edge occurs at most once.

\(^3\)In some other texts, this is called an Euler circuit.

\(^4\)More precisely, a graph \(G\) is weakly connected iff there is a path from any vertex to any other vertex in the graph \(H\) with

\[ V(H) = V(G), \text{ and} \]

\[ E(H) = E(G) \cup \{(v \to u) \mid (u \to v) \in E(G)\}. \]

In other words \(H = G \cup G^{-1}\).
(b) Suppose that a trail in a weakly connected graph does not include every edge. Explain why there must be an edge not on the trail that starts or ends at a vertex on the trail.

Solution. If an edge not on the trail starts or ends on it, then that already is the desired edge. So suppose there’s an edge, $e$, not on the trail that neither starts nor ends at a vertex on the trail. Since $G$ is weakly connected, there is a path, $p$, in $G \cup G^{-1}$ from any vertex, $v$, on the trail to an endpoint of $e$. Then the first edge in $p$ that is not on the trail will be the desired edge of $G$.

In the remaining parts, assume the graph is weakly connected, and the in-degree of every vertex equals its out-degree. Let $w$ be the longest trail in the graph.

(c) Show that if $w$ is closed, then it must be an Euler tour.

Hint: part (b)

Solution. Suppose $w$ is closed and some edge is not on $w$. By part (b), there must be an edge, $e$, not on $w$ that starts or ends on $w$. If $e$ starts on $w$ at some vertex, $v$, then since $w$ is closed, we may assume that it starts and ends at $v$. It follows that $w \hat{v} e$ a trail that is longer than $w$ by one edge, contradicting its maximality. Similarly, if $e$ ends on $w$ at some vertex, $v$, then the walk $e \hat{v} w$ is a longer trail, again contradicting maximality.

So no edge can be missing from $w$.

(d) Explain why all the edges starting at the end of $w$ must be on $w$.

Solution. Otherwise we could extend $w$ to a longer trail with any edge leaving the end not already in $w$.

(e) Show that if $w$ was not closed, then the in-degree of the end would be bigger than its out-degree.

Hint: part (d)

Solution. Let $v$ be the end vertex of $w$. Given that $v$ is not the start of $w$, it follows that, other than at the end, any occurrence of $v$ in $w$ is preceded by an edge that ends at $v$ and is followed by an edge that leaves $v$. Since $w$ is a trail, all the edges are distinct among these pairs of edges at each non-end occurrence $v$. In addition, the final edge in $w$ ends at $v$ and is distinct from all the paired edges. Altogether, this implies that there are an equal number of edges in $w$ that enter $v$ and that leave $v$, except for the last edge, which enters $v$ and so adds 1 more to the in-degree. So there is one more edge in $w$ entering $v$ than leaving $v$. But by part (d), all edges leaving $v$ are in $w$, proving that $\text{indeg}(v) \geq 1 + \text{outdeg}(v)$.

(f) Conclude that if the in-degree of every vertex equals its out-degree in a finite, weakly connected digraph, then the digraph has an Euler tour.

Solution. If all vertices in $G$ have equal in- and out-degree, then by part (e), the only possibility is that the end of $w$ equals the start, that is, $w$ is closed. So by part (c), $w$ is an Euler tour.