Fifteenth Lecture: 9/5

De Bruijn graphs. Let $\mathcal{A}$ be a finite set and k a positive integer. The elements of the k-fold Cartesian product $\mathcal{A}^k$ can be identified with strings $a_1a_2\dots a_k$, $a_i \in \mathcal{A}$. In this situation it is common to speak of $\mathcal{A}$ as an *alphabet* and of the strings as *words of length* k in the alphabet $\mathcal{A}$.

A digraph $G = (V,\, E)$ is called a *De Bruijn graph* if there is a finite alphabet $\mathcal{A}$ and a positive integer k such that

(i) V consists of all words of length k in the alphabet $\mathcal{A}$,

(ii) there is a directed edge from the word $a_1a_2\dots a_k$ to the word $a_1'a_2'\dots a_k'$ if and only if $a_i' = a_{i+1}$ for all $i = 1,\, 2,\, \dots,\, k-1$.

Note that (ii) implies that there will be a loop at each vertex of the form $aa\dots a$, $a \in \mathcal{A}$. If we ignore the loops then, for a vertex $\boldsymbol{v} = a_1a_2\dots a_k$ one has

$$\text{Outdeg}(\boldsymbol{v}) = \text{Indeg}(\boldsymbol{v}) = \begin{cases} |\mathcal{A}| - 1, & \text{if } a_1 = a_2 = \cdots = a_k, \\ |\mathcal{A}|, & \text{otherwise.} \end{cases}$$

Hence, from Theorem 14.6(ii) it follows that a De Bruijn graph always possesses an Euler cycle. Note that this applies even if we include the loops - we can imagine performing an Euler cycle in the loopless graph and executing each loop the first time we visit the corresponding vertex.

Example 15.1. (The Keycode Problem) In Sweden, apartment buildings are usually equipped with electronic door locks and to get into the building one must punch in the correct sequence of four decimal digits. Usually it is the case that it suffices to punch in the correct four digits consecutively. So, for example, if the code is 1234 and you begin by erroneously punching 121, then it suffices to continue with 234 to gain entrance, you don't need to "start all over" and punch in the 1 again. This feature means that a robber, seeking to gain entrance but who has no clue what the correct code is, does not in the worst case (for him !) need to punch in $4 \times 10^4 = 40,000$ digits to be absolutely sure of gaining entrance. Indeed, a sequence of just $10^4 + 3 = 10,003$ digits contains 10^4 different codes, so the question arises whether there exists such a sequence of $10^4 + 3$ digits which includes every 4-digit code exactly once (and thus makes the robber's job easier by a factor of 4) ?

The answer is yes ! For consider the De Bruijn graph whose nodes are words of length 3 in the alphabet $\{0,\, 1,\, \dots,\, 9\}$. Every directed edge in this graph corresponds to a 4-digit code and it is easy to see that an Euler cycle, including the 10 loops, corresponds to a sequence of $10^4 + 3$ digits which includes every 4-digit code exactly once.

We can "see" how this works by taking a simpler example, say $\mathcal{A} = \{0,\, 1\}$ and $k = 2$. The De Bruijn graph is shown in Figure 15.1(i). It contains $|\mathcal{A}|^k = 2^2 = 4$ nodes. The sequence of edges in an Euler cycle (found via the usual greedy search) is shown in Figure 15.1(ii). The corresponding sequence of $|\mathcal{A}|^{k+1} + |\mathcal{A}| = 2^3 + 2 = 10$ binary digits is

$$0001110100$$

and one may check that this includes each of the $2^3 = 8$ three-digit binary words exactly once.

Definition 15.2. A path in a graph $G = (V, E)$ is called a *Hamilton path* if it visits every vertex exactly once. A path which visits every vertex exactly once and then returns via an edge to the starting vertex is called a *Hamilton cycle*.

The problem of deciding whether or not a graph contains a Hamilton path or cycle is known to be much more difficult than the corresponding problem for Euler paths/cycles, which we resolved completely in the previous lecture. Indeed, it is one of the oldest known examples of a so-called *NP-complete problem*. It is beyond the scope of this course to explain what this means, but it is a central notion in the subject of *complexity theory (of algorithms)* and one reason why the subject of graph theory is so important for theoretical computer science is that it is a rich source of concrete NP-complete problems - we will be seeing further examples in the coming lectures. Philosophically, large classes of "difficult" algorithmic problems can be encoded as problems in graph theory, and the decision problem for Hamilton paths/cycles is a classic example.

A common way of popularising the decision problem for Hamilton paths/cycles is to consider it as a special case of the *Travelling Salesman Problem*. Here one thinks of the nodes in a graph as cities and the edges as representing those pairs of cities for which there exists a flight connection. The travelling salesman wishes to visit every city but has no reason to visit a place more than once, if he can avoid it. Whether or not he can achieve his goal is equivalent to asking if the graph has a Hamilton path and, assuming he'd like to end up back home where he started, whether it has a Hamilton cycle[1].

Intuitively, one can see why the decision problem for Hamilton paths is harder than the corresponding problem for Euler paths, by considering that a Hamilton path in an n-vertex graph uses $n - 1$ edges, while such a graph can in principle have anything from 0 to $\frac{n(n-1)}{2}$ edges. Thus, for most graphs, a Hamilton path would use only a small fraction of the total number of available edges, though not a negligible fraction, we still have to use $n-1$ edges after all[2]. In contrast, an Euler path must use every edge exactly once. This is a very stringent requirement, which leads to a very sharp (and restrictive !) characterisation of those graphs for which it is possible.

This intuition would, however, naturally lead one to expect that, the *denser* the graph, by which we mean the greater the quotient $|E|/\binom{n}{2}$, the greater the likelihood that Hamilton paths or cycles exist. Indeed, in the extreme case, consider K_n. This possesses Euler cycles if and only if $n \geq 3$ is odd, by Theorem 14.4(ii). On the other hand,

[1] In the full TSP, each edge e comes equipped with a non-negative weight $w(e) \in \mathbb{R}_+$, representing the cost of the flight between those two cities. The problem is then to find a path in the graph which visits every city at least once and for which the total cost is minimised. Our special case above is to set $w(e) = 1$ for every e and ask if there exists a path of total cost $|V| - 1$, or a cycle of total cost $|V|$. We will be returning to weighted graphs from Lecture 17 onwards.

[2] A more rigorous way of developing this intuition is to firstly imagine the graph being chosen randomly by inserting each of the $\binom{n}{2}$ possible edges with probability $1/2$, independent of all other edges, and then to search for a Hamilton path by taking a "random walk" from a randomly chosen starting vertex. It's beyond the scope of the course to delve into this further.

for each $n \geq 3$, K_n possesses $\frac{n!}{n} = (n-1)!$ Hamilton cycles, since every permutation of the n vertices corresponds to a Hamilton path and there are n possible starting points for a given cycle.

However, high density on its own is not enough to guarantee Hamilton paths. Consider, for example, a graph which is the union of K_{n-1} and an isolated vertex. It contains $\frac{n-2}{n-1}$ of all possible edges but obviously no Hamilton path. This might suggest that, in addition to having lots of edges we would like them to be "spread evenly around". There is, in fact, a theorem which makes this precise:

Theorem 15.3. (Dirac's Theorem) *Let $G = (V, E)$ be a graph with $|V| = n > 2$. If $deg(v) \geq n/2$ for every $v \in V$, then G possesses a Hamilton cycle.*

Proof. The proof is by contradiction. Fix an $n > 2$ and suppose the theorem is false for this value of n, in other words, suppose there is an n-vertex graph which contradicts the theorem. Then there must be such a graph with the maximum possible number of edges. Pick any such graph and call it G. Thus we're assuming that

(i) $\deg(v) \geq n/2$ for every $v \in V(G)$,

(ii) G possesses no Hamilton cycle,

(iii) adding any edge to G will create a Hamilton cycle.

We will have a contradiction if we can prove that G had a Hamilton cycle all along. We start by using (iii). Pick a pair of vertices x, y such that the edge $\{x, y\}$ is not in G. Adding it must create a Hamilton cycle and we may assume any such cycle includes the edge $\{x, y\}$, as otherwise it would already have been present in G. So we can pick such a cycle and let x be the "first" and y the "last" vertex, i.e.: the cycle reads

$$v_1 = x \to v_2 \to v_3 \to \cdots \to y = v_n \to x.$$

Now define the subsets S and T of $\{1, 2, \ldots, n-1\}$ as follows:

$$S = \{i : \{x, v_{i+1}\} \in E(G)\},$$
$$T = \{i : \{v_i, y\} \in E(G)\}.$$

Note that $|S| = \deg(x)$ and $|T| = \deg(y)$. Hence, $|S| \geq n/2$ and $|T| \geq n/2$. But both are subsets of $\{1, 2, \ldots, n-1\}$, so $|S \cup T| \leq n-1$. It follows that $S \cap T \neq \phi$. Let $i \in S \cap T$, so both $\{v_i, y\}$ and $\{x, v_{i+1}\}$ are edges in G. We can now construct a Hamilton cycle in G as follows (see Figure 15.2):

$$v_1 = x \to v_2 \to \cdots \to v_i \to y = v_n \to v_{n-1} \to \cdots \to v_{i+1} \to x.$$

This is a contradiction, completing the proof. □

Remark 15.4. **(i)** The theorem doesn't hold for $n = 2$, since K_2 satisfies the requirement that every vertex has degree at least $2/2 = 1$, but obviously it has no Hamilton cycle (though it does have a Hamilton path).

(ii) Dirac's theorem gives a *sufficient* condition for existence of a Hamilton cycle, but it's a million miles away from being a *necessary* one. For example, the cycle C_n *is* a Hamilton cycle for every $n \geq 3$, but doesn't satisfy Dirac's condition once $n \geq 5$.

Paths and the adjacency matrix. Before leaving the subject of paths and cycles for a while, I want to just mention a rather cute application of linear algebra in this area. We quote it as a theorem:

Theorem 15.5. *Let $G = (V, E)$ be a graph on n labelled vertices $1, 2, \ldots, n$, let $A = A_G$ be the corresponding $n \times n$ adjacency matrix and let k be a positive integer. Let $a^k_{i,j}$ denote the (i, j):th entry of the matrix A^k. Then $a^k_{i,j}$ is the number of paths of length k in G from vertex i to vertex j.*

Proof. Simple, but a bit messy to write out and left to the interested reader as an exercise. Basically it's just a matter of unwinding the definition of matrix multiplication, as applied to the adjacency matrix. □

Note that the adjacency matrix A is symmetric, hence diagonalisable. Therefore, the theorem implies that computing the number of paths of a given length between a pair of vertices essentially reduces to finding the eigenvalues of the adjacency matrix. For digraphs, A is no longer symmetric, which makes the problem a bit trickier. Indeed, if we allow multigraphs and loops, then A can be any $n \times n$ matrix whatsoever.