Competitive Programming and Mathematics Society

Mathematics Workshop

Graph Theory (with Linear Algebra)

Zac (with Cyril)

Table of contents

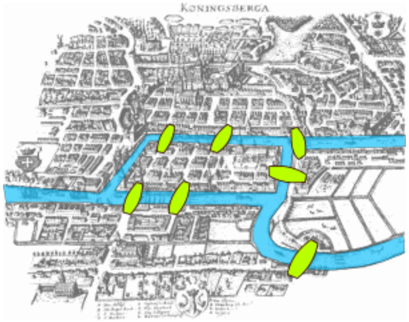
Welcome

- Next mathematics workshops in week 7?
- Slides will be uploaded on website (unswcpmsoc.com)

Attendance form

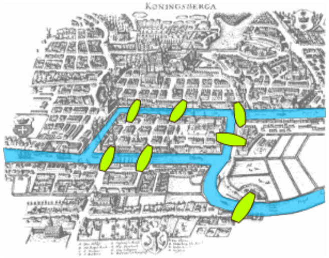
A Classic Problem

- Funny hat man (Leonhard) was in Königsberg, wishing to visit all the landmasses and bridges.
- For peak efficiency we want to visit each bridge only once. Can this be done?



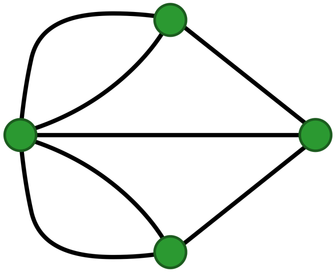
A Classic Solution

- Each time we land on a landmass, we must leave it, unless it is our final destination.
- Each time we leave a landmass, we must have landed on it from a bridge, unless we started there.
- Since there is only one start, and one end, at most two landmasses can be connected to an odd number of bridges attached to them.
- All the rest need an even number of bridges (by pairing entry bridges to exit bridges).
- Since the diagram shows all landmasses have an odd number of bridges, Leonhard cannot fulfil his dreams, and thus is sad.



Graphs

- A graph is an object where one set of objects, vertexes, are joined together in some arrangement by edges.
- In the previous problem, our vertexes were landmasses, and they were joined by bridges acting as edges.
- A simplified diagram of the Königsberg graph is shown below.
- See problem sheet question 6.



Common Themes - Graph Invariants

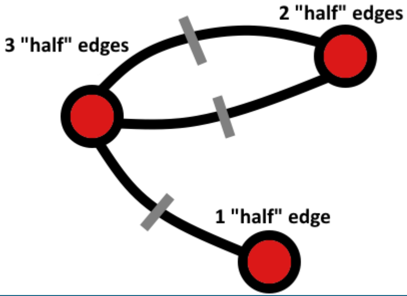
- A common problem may be the need to create a graph with certain properties, such as being traversable by the method from before, or asking if a certain graph has said properties, perhaps by comparing it with other graphs.
- We also may want to know about specific kinds of graphs.
- We call a graph "simple" if each edge joins two different vertices, and no two vertices have more than one edge between them.

Example problem

- Does there exist a graph with four vertices, three of which have 3 edges joined to them, the fourth having 2 edges joined to it?
- Note: if one edge connects a vertex to itself, we call it a "loop", and it counts as joining twice to that vertex.

Example problem - solution

- Does there exist a simple graph with four vertices, three of which have 3 edges joined to them, the fourth having 2 edges joined to it?
- Lets count the number of edges here: each of 3 vertices has 3 "half-edges" in some sense, as an edge joins to two vertices, and then a 4th vertex has 2 "half-edges", giving a total $3 * 1.5 + 2 * 0.5 = 5.5$ edges, so this graph cannot exist.
- In general, notice that each edge connects to two vertices, so the total number of edges joined to by vertices must be even, as each edge is double counted.

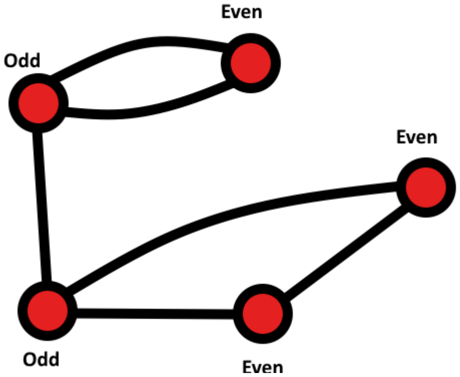


Vertex Degree Properties

- We call the number of edges connected to a vertex that vertex's degree.
- As established, the total degree of graph is even. i.e. degree $= 0 \bmod 2$
- If we think about vertexes with even degree, just considering them we get an even contribution to the total degree of the graph.
- Proof: if vertexes $x_1, x_2, ..., x_n$ each have even degree $2y_1, 2y_2, ..., 2y_n$, the sum of their degrees is $2(y_1 + y_2 + ... + y_n) = 0 \bmod 2$
- So we can also deduce that the contribution from vertices with an odd degree is even. If these vertexes are $z_1, z_2, ..., z_n$ with degrees $2w_1 + 1, 2w_2 + 1, ..., 2w_n + 1$, their total degree is $2(w_1 + w_2 + ... + w_n) + 1 \times$# of vertices. For this to be even, # of vertices must be even.
- Thus we have derived that in general, in a simple graph, the number of odd degree vertices must be even, which provides a general solution to problems similar to the last one.

Vertex Degree Properties Example

- Consider the graph below. Notice that the even degree vertices contribute 6 to the total degree, which is even. Since the odd vertices each contribute an odd amount to the total degree, we need an even number of them to keep the total degree even, and indeed we do have 2 odd vertices.



Graph Walks

- Suppose we have a graph G. How many ways are there to travel along edges to get between two connected points?
- Note: here, connected means that there is at least one way to get between the points.

Graph Walks

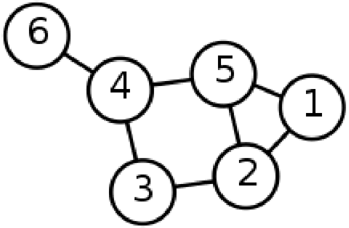
- Suppose we have a graph G. How many ways are there to travel along edges to get between two connected points?
- Note: here, connected means that there is at least one way to get between the points.
- Solution: An infinite amount!
- For example, since the points are connected, I can just travel along this walk $2n+1$ times and always end on my desired point.
- So maybe a more interesting question is how many ways can I travel between two points, using at most n steps.
- Note: here, by n steps, we mean traversing across n edges in total.

Graph Walks

- How many ways can I travel between two points in graph G, using at most n steps?
- This is equivalent to asking how many ways I can do it in exactly k steps, and then summing the answers from $k = 1$ up to n.
- Well then firstly, how many ways can I travel between two points in a graph in 1 step?

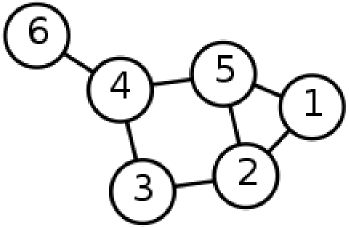
Graph Walks

- How many ways can I travel between two points in graph G, using at most n steps?
- This is equivalent to asking how many ways I can do it in exactly k steps, and then summing the answers from $k = 1$ up to n.
- Well then firstly, how many ways can I travel between two points in a graph in 1 step?
- Solution: Exactly how many edges join those vertices! For example in the graph shown here we can travel between vertexes 4 and 5 with 1 step only 1 way, as they are joined by 1 edge.



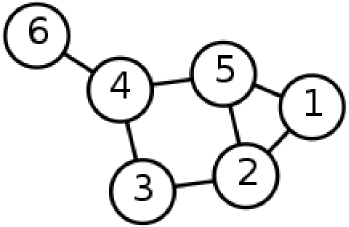
Graph Walks

- How about 2 steps? Hopefully we can establish an inductive pattern that relies on taking 1 step twice.
- Don't worry about a nice solution, just think about how given a graph, say the one below, we could count all 2 step walks between vertices, e.g. vertices 2 and 4.



Graph Walks

- How about 2 steps? Hopefully we can establish an inductive pattern that relies on taking 1 step twice.
- Don't worry about a nice solution, just think about how given a graph, say the one below, we could count all 2 step walks between vertices, e.g. vertices 2 and 4.
- Solution: we keep track of all vertices that can be reached by 1 step, and then find how many of those can reach 4 with 1 step and sum them up! So here 3, 5, and 1 can be reached from 2 in 1 step in 1 way each, but only 3 and 5 can reach 4 in 1 step in 1 way, so the number of 2 step walks from 2 to 4 is $1 + 1 = 2$.

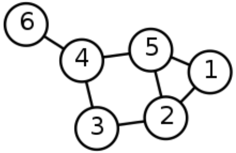


Graph Walks

- xcSo far we have described a lengthy process that seems brute-forced and difficult to remember.
- We are taking the number of edges between the starting node and each other node, and then for each of those nodes we will multiply the number of ways to get to the ending node, and then sum the total.
- Note: possibly some extra thought here is required for why we need to specify multiplication, and it relies on having extra edges between nodes, but accounting for this case is useful later.
- Thinking of alternative representations of things in math is always good, so maybe since we have a lot of numbers to keep track of, we can store them in vectors.

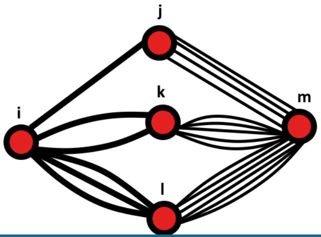
Adjacency Vectors

- Let v_{2-} be the vector of the number of edges joining vertex 2 to each other vertex, and v_{-4} the vector of the number of edges joining each vertex to vertex 4. For example again in this graph 2 joins to vertexes 1, 3, and 5 once each, and 0 times to vertexes 2, 4 and 6 have: $v_{2-} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$. Also $v_{-4} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{pmatrix}$
- We now notice our calculation from before is the dot product of these two, $v_{2-} \cdot v_{-4} = 0 + 0 + 1 + 0 + 1 + 0 = 2$.



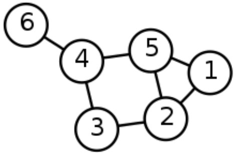
Adjacency Vectors, Alternative Example

- For this other graph, there are 1, 2, and 3 ways to get from vertex i to j, k, and l, and 4, 5, and 6 ways to get from vertices j, k, and l to vertex m. Using adjacency vectors with components in the order i, j, k, l, and then m we have:
- $v_{i-} = \begin{pmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 0 \end{pmatrix}$. Also $v_{-m} = \begin{pmatrix} 0 \\ 4 \\ 5 \\ 6 \\ 0 \end{pmatrix}$. $v_{i-} \cdot v_{-m} = 0 + 4 + 10 + 18 + 0 = 32$.
- So there are 32 length 2 walks from i to m.



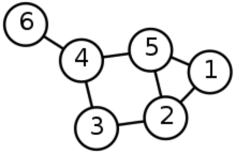
Adjacency Vectors

- Using vectors may seem strange, but the benefit of this becomes clearer once we look at longer walks of length k. If we know how many ways there are to get from our starting vertex to every vertex in the graph in $k-1$ steps, again stored in a vector, then taking the dot product with our adjacency vector v_{-end} from before gives us the number of ways to get to the end in k steps.
- This means that as we calculate larger and larger walks, we really want to calculate the walks between any two pairs of vertices, because they seem necessary for the later calculations.
- Let us do an example by listing out all adjacency vectors for this graph.



Adjacency Matrices

- Let us do an example by listing out all adjacency vectors for this graph.
- We store these vectors in a list, forming a 2D array called a "matrix". Often, the matrix will share names with the graph, so here we will call it G.
- $G = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$



- Notice that the entry at the ith row and the jth column (ijth entry) gives the number of edges between vertexes i and j. Because the graph is simple, all entries are 0 or 1 and all "main diagonal" entries are 0. We can spot v_{2-} from before is the 2nd column, and v_{-4} is the 4th. The graph is symmetric because edges from i to j are also from j to i, so v_{2-} is also the 2nd row.

Matrix-Vector Multiplication

- If we want to see all the length 1 walks between vertexes i, j we look at the ijth entry in the adjacency matrix. For all length 2 walks between a particular vertex k and vertex j, we need to take the dot products of each adjacency row vector in the matrix with the adjacency vector v_k. An example is shown for $k = 2$.
- $Gv_k = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \\ 0 \\ 2 \\ 1 \\ 0 \end{pmatrix}$
- For example, entry 2 of the output vector is the dot product of the 2nd row with the column vector, and $1 * 1 + 0 + 1 * 1 + 0 + 1 * 1 + 0 = 3$.

Matrix-Matrix Multiplication

- Recall that for length k walks we wanted knowledge about walks between any two vertices of length $k-1$, so we need to multiply this matrix by all adjacency vectors. An object we established stored all these vectors was the matrix itself, so we now define matrix-matrix multiplication.
- $G \times G = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$
- Here our output column vectors will be the vectors we would have gotten by doing matrix-vector multiplication from before with just one column of the second matrix.
- We can demonstrate the ijth entry of the new matrix will be the dot product of the ith row of the first matrix with the jth column of the second.

Matrix-Matrix Multiplication

- Let us demonstrate this computation with a smaller example.
- In some applications, matrices are no longer symmetric, so we specify this kind of multiplication uses the rows of the first matrix, and the columns of the second matrix.
- $\begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} \begin{pmatrix} 4 & 5 \\ 6 & 7 \end{pmatrix} = \begin{pmatrix} 0*4+1*6 & 0*5+1*7 \\ 2*4+3*6 & 2*5+3*7 \end{pmatrix} = \begin{pmatrix} 6 & 7 \\ 26 & 31 \end{pmatrix}$
- The ijth entry of this new matrix will be the dot product of the ith row of the first matrix with the jth column of the second.

Length k Walks

- $G^2 = \begin{pmatrix} 2 & 1 & 1 & 1 & 1 & 0 \\ 1 & 3 & 0 & 2 & 1 & 0 \\ 1 & 0 & 2 & 0 & 2 & 1 \\ 1 & 2 & 0 & 3 & 0 & 0 \\ 1 & 1 & 2 & 0 & 3 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$
- Now this matrix gives us the length 2 walks from i to j in the ijth entry.
- To get the length 3 walks to vertex k, we multiply by v_k, the adjacency vector, and the output vector gives length 3 walks from each vertex to vertex k. Doing this for all vectors involves multiplying again by G.
- This is now getting quite complicated, and we should look back over everything we have done, but from this point we can begin to see an amazing fact:
- The ijth entry of G^k gives the number of walks between i and j of length k.

Length k Walks

- Good skeptics at this point may point out that this is a lot of adding and multiplying, and that this is still a brute forced solution, even if it is now written possibly more compactly.
- Thankfully, there are actually some magical methods to calculate G^k generally in many cases much faster than by typical matrix-matrix multiplication.
- For more information on this in particular you can look up diagonalisation of matrices, or if you are feeling especially cheeky Jordan Block matrices.
- Another method to getting large matrix powers is to square the matrices again and again, i.e. $A^{2^n+k} = A^{2^{2^{\dots 2}}}AA\dots A$ with n squarings and then k regular multiplications. Similar more efficient algorithms can be made.

Walks with lengths $\leq n$

- Back to the original question however, we have so far established that the number of length k walks in a graph between points i and j is given by the ijth entry in the kth power of the adjacency matrix, A^k.
- As a sum now, all walks $\leq n$ would be given by the ijth entry of the matrix $A^n + A^{n-1} + ... + A^2 + A = \sum_{k=1}^{n} A^k$
- Note that when summing together these matrices, much like vectors each entry of the matrix is added to the corresponding entry of the next matrix, so indeed at the end our final matrix accounts for all walks of length $\leq n$ between vertexes i and j at entry ij.

Where from here?

- We have really just introduced graphs, and established how to use linear algebra to analyse one major problem, what else can you look at?
- Graph traversals: Euler paths, Hamiltonian paths
- Special types of graphs: bipartite, trees, directed (cyclic, acyclic), weighted
- Graph topology: isomorphisms, enumeration
- Algorithms on graphs: Dijkstra's algorithm, colouring algorithms

Attendance form :D

Further events

Please join us for:

- Maths workshop in two weeks
- Social session tomorrow
- Programming workshop next week