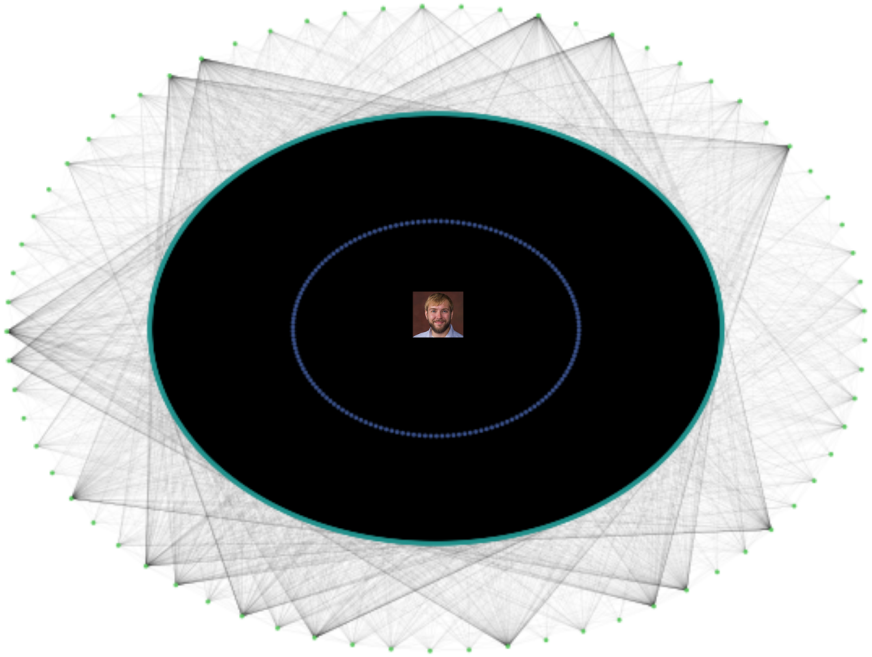
Academic connectivity of Harvard students

Calculating Dusty score with Linear Algebra

Sirui Cai, Je Qin Chooi



Math 22a Final Project

Harvard University
November 2022

Cover Letter

Within this project, Sirui worked on Sections 1, 2 and half of Section 3. Je Qin worked on the second half of Section 3, and Sections 4 and 5.

We are extremely grateful for the thoughtful feedback and advice that we received from our TF, Eunice Sukarto, and our peer reviewers, Hugh Hankenson, Ezra Kizito, Hans Bach-Nguyen, Ashley Redhead, Layla Dawit and Thomas Gustafson. Some recurrent suggestions that we received included to illustrate definitions of graph theory terminology more clearly, to omit some redundant proofs of self-evident theorems/observations, and to reconsider the relevance of Section 4, which discusses computational challenges.

To address the first point, we revised the phrasing of our definitions in Section 2.1, and included Figure 1 as a visual aid to help readers better visualize what the various terms represent. We also made a more deliberate effort to clarify the relationship between the distance matrix and the APSP solution matrix (namely, that the solution to the APSP problem is represented by the distance matrix), and streamlined our references to this particular matrix in later parts of the paper.

Noting the feedback we received on distinguishing theorems from observations, and omitting unnecessary proofs, we changed Theorem 3.2 into an observation and shortened its proof to a brief intuitive explanation. We also removed proofs for the theorems in Section 4.1.

A few of our reviewers also pointed out that they were unclear about how Section 4 fits into the rest of our paper. We are convinced that the results here are significant (since without the result from the spectral theorem, it wouldn't have been physically possible to compute the Dusty score for the class of 2026 within the time frame given for this assignment). Furthermore, we thought it was a really cool application of the spectral theorem. So, we decided to retain the section, but made sure to be more explicit about explaining its relevance to our paper.

Finally, the most exciting addition to our final draft is the results of our actual computation. We are deeply grateful to the FAS Registrar's Office for kindly providing us with the data that made this computation possible. We found that apart from the 158 first-year students in Math 22A, the vast majority of first-year students had a Dusty score of 2, with only 18 students having a Dusty score of 3. We also generated a set of class enrollment data for a sample of 1000 Harvard students (all class years) based on class sizes reflected in the Q Guide for 2021 Fall. The class enrollment profile of each student was generated randomly based on the probability of being in a certain class (depending on its size). Comparing the Dusty score distribution from the actual student sample and our randomly generated data confirmed intuitive understandings of class selection patterns: the average Dusty score was lower in the actual freshman sample, which makes sense because we would expect there to be a higher degree of class connections when a population clusters around a set of classes (in this case, freshman-oriented classes) instead of being randomly spread across all the classes offered by the college.

Overall, it was really interesting to see how linear algebra could be applied to graph theory to study social networks in our immediate real-world context, and we hope that learning more about this area of application was as enjoyable for our readers as it was for us.

1 Introduction

Imagine having any Harvard student walk up to you, and being able to tell their academic connectedness with every other student. We explore this aspect of network connectivity through the Dusty score. Defined as the distance from Dusty in a graph where students are connected to one another by means of being in the same classes, the Dusty score measures a student's degree of separation from Dusty based on shared class connections. For example, a student who is directly taught by Dusty has a Dusty score of 1, while a student who is not taught by Dusty but is in the same class as someone who is will have a Dusty score of 2. Are there algorithms that can calculate the Dusty score of a given student population?

In *Graph Algorithms in the Language of Linear Algebra*, Jeremy Fineman outlines the Bellman-Ford and Floyd Warshall algorithims, which can achieve this goal by solving the Single-Source Shortest Path (SSSP) and All-Pairs Shortest Path problems respectively (APSP) (defined in Section 2). [3] This paper explores whether we can derive an *original* algorithm, based on a series of matrix operations, that takes input data in the form of the classes taken by each student in a community and calculates each of their Dusty scores as the output.

In Section 2, we will define the graph that will be used to model student-class networks. Section 3 will then propose a series of matrix operations that can be used to calculate the Dusty scores of a student population given their class enrollment data. In Section 4, we will explore results in linear algebra that can simplify the computational process. Finally, in Section 5, we will apply the process outlined in Section 3 to calculate the actual Dusty scores of students in the Harvard Class of 2026 using available class enrollment data. The results of the calculations will yield answers to questions such as what the average Dusty score of the Harvard Class of 2026 is, and how the number of students and classes offered affects the extent of connectivity of a college community.

Our project is a subset of the more general Single-Source Shortest Path (SSSP) problem in graph theory. Calculating the Dusty scores of a student body can be generalized to finding the distance of the shortest path from a particular point on a graph (Dusty in our case) to every other point (students in our case). The SSSP problem has wide-ranging applications; some of the more prominent use cases can be found in the fields of transportation and telecommunication. [8]

2 Modelling the student-class network

Before tackling the motivating questions, we will first set up the graph that will be used to model student-class networks.

2.1 Basic definitions in graph theory

A **graph** is defined as a an ordered pair $G = (V, E)$ comprising a set V of vertices (hereafter referred to as nodes) and a set E of edges. A **node** is essentially a point on the graph. An **edge** is a pair of nodes (can be thought of as a line joining two nodes). In our paper, we will be dealing primarily with **unordered** edges, in which the edge simply joins two nodes without pointing from one to the other. An **undirected graph** comprises only unordered

edges. Two nodes are **adjacent** if they are directly connected by an edge. Edges can be assigned a numerical value, called its **weight**, representing the distance between two nodes.

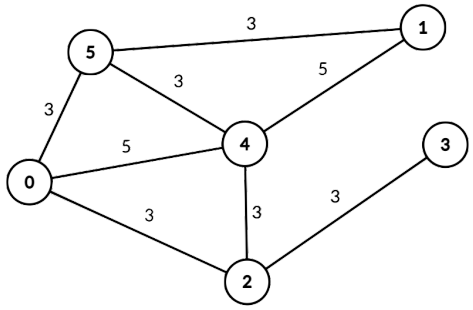


Figure 1: Undirected graph in which nodes 0-5 are connected to each other via unordered edges of weights either 3 or 5

A **walk** is a sequence of edges which joins a sequences of nodes. A walk can also be thought of as an alternating sequence of nodes and edges. Using the graph in Figure 1, $0 \to 5 \to 0 \to 2$ is a walk of distance $3 + 3 + 3 = 9$ units. A **path** is a walk where all edges and nodes in the sequence are distinct. For instance, $0 \to 4 \to 2$ and $0 \to 2$ are paths from node 0 to 2, but $0 \to 5 \to 0 \to 2$ is not, since this walk passes through node 0 twice. The **shortest distance** between two nodes is the lowest possible sum of the weights of the edges involved in any path between them. The shortest distance from node 0 to node 2 is thus 3 units, which is obtained using the path $0 \to 2$. The above definitions were derived from [6].

Observation The length of the shortest walk between two nodes is the same as the length of the shortest path between nodes.

The **Single-Source Shortest Path (SSSP)** problem involves finding the shortest path between a given node and all other nodes in the graph. The **All-Pairs Shortest Path (APSP)** problem involves finding the shortest path between all pairs of nodes. The solution to the APSP problem can be expressed in a matrix, which we will define as follows:

Definition The **distance matrix** is a matrix such that the (i,j) entry is the shortest distance between i and j. This is analogous to the APSP solution matrix.

2.2 Deriving the student-class network model

In the context of our paper, we are working with a graph representing how every student in the Harvard Class of 2026 is connected to one another through having shared classes. Each student is represented by a node on the graph. If students A and B are in the same class, then their nodes are connected by an **unordered edge**. All edges here are **equally weighted** with a value of 1. This means that we are working with an **undirected**, **unweighted** graph. These properties are important to note as they make it possible for the theorems discussed later in this paper to apply.

3 Calculating the Dusty score

By modelling the students as nodes and classmate relationships as edges, we can now define the Dusty score.

Definition A student's **Dusty score** is their shortest distance to Dusty.

Note that Dusty has a unique Dusty score of 0. To calculate the Dusty score of all students is then to calculate the shortest distance of all students to Dusty. This is a **Single-Source Shortest Path (SSSP)** problem. In the following subsections, we will show that we can solve the more general case of the **All-Pairs Shortest Path (APSP)** problem with linear algebra. The SSSP solution is then the entries on the row corresponding to that single source. For the Dusty score, we read off the entries on Dusty's row. Before delving into the matrices required as a preparation for the APSP solution, we can already note that all students in Math 22 have a Dusty score of 1, since they have a direct edge to Dusty.

3.1 The membership matrix

Let there be n students and m classes. Our raw data will come in the form of class enrollments for each student. This data can be represented by what we define as a membership matrix:

Definition The **membership matrix** is an $n \times m$ matrix where (i, j) entry is '1' if student i takes class j and '0' otherwise.

Note that each row represents an individual student and the columns represent classes. Here is an example where the number of students $n = 6$, the number of classes $m = 9$ and each student has to take 3 classes. For convenience, we set Dusty as "Student 1".

	MATH 22A	PHYSICS 15A	COMPSCI 50	ECON 10A	MATH 21A	GENED 1194	ECON 1011A	COMPSCI 61	MATH 55A
Dusty "Student 1"	1	0	0	0	0	0	0	0	0
Student 2	1	0	1	1	0	0	0	0	0
Student 3	1	0	1	1	0	0	0	0	0
Student 4	0	1	1	0	1	0	0	0	0
Student 5	0	1	0	0	1	1	0	0	0
Student 6	0	0	0	0	0	0	1	1	1

The corresponding membership matrix is then the entries in the table. Note that each row has exactly three '1's (corresponding to the exact number of classes each student must take) except for Dusty, who in our case is only a member of the column MATH 22A.

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \tag{1}$$

3.2 The adjacency matrix

Obtaining the adjacency matrix of the student-class graph is a good starting point for deriving the distance matrix. We thus want to find a way to derive the adjacency matrix from the membership matrix.

Definition For a graph with m nodes, the **adjacency matrix** A is an $n \times n$ matrix where the (i, j) entry is the number of direct edges from node i to node j.

Observation The adjacency matrix of an undirected graph is symmetric.

For an undirected graph, the number of direct edges from i to j is equal to the number of direct edges from j to i. Hence, the (i, j) entry is equal to the (j, i) entry, and the matrix is symmetric.

In the context of our paper, the (i, j) entry of the adjacency matrix is the number of classes shared by student i and student j.

Theorem 3.1 Multiplying the membership matrix M by its transpose M^T yields the adjacency matrix A.

Using the same example as above, we obtain the adjacency matrix A for the sample from its membership matrix M as follows:

$$A = MM^T = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 3 & 3 & 1 & 0 & 0 \\ 1 & 3 & 3 & 1 & 0 & 0 \\ 0 & 1 & 1 & 3 & 2 & 0 \\ 0 & 0 & 0 & 2 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 \end{bmatrix} \tag{2}$$

Proof. Define the row vector m_i as the ith row of the membership matrix M and the column vector m_i^T as the transpose of m_i.

$$M = \begin{bmatrix} — & m_1 & — \\ — & m_2 & — \\ & \vdots & \\ — & m_n & — \end{bmatrix}, \qquad M^T = \begin{bmatrix} | & | & & | \\ m_1^T & m_2^T & \dots & m_n^T \\ | & | & & | \end{bmatrix}$$

We assume that $A = MM^T$ and then proceed to show that A fulfils the definition of the adjacency matrix. By the properties of matrix multiplication, the entry a_{ij} of A is the result of the vector multiplication of m_i by m_j^T. We note that m_i and m_j^T are both vectors comprising only '0's and '1's, and the kth entry of each of these vectors is '1' if and only if student i or j respectively takes the kth class. Hence, the kth term in the sum from the vector multiplication is 1 if and only if students i and j both take the kth class and 0 otherwise. The result of the vector multiplication, being the sum of these n terms, is thus the number of classes that students i and j share. Since the entry a_{ij} is the number of classes shared by students i and j, we arrive at the original definition of the adjacency matrix. □

Note that the diagonal entries except for the first Dusty row are all 3, which is the number of classes taken by the students because it is necessarily the number of classes that each student has in common with themselves. The adjacency matrix allows us to represent the network as a graph (shown below) by drawing the number of edges between two nodes according to the entries of the matrix.

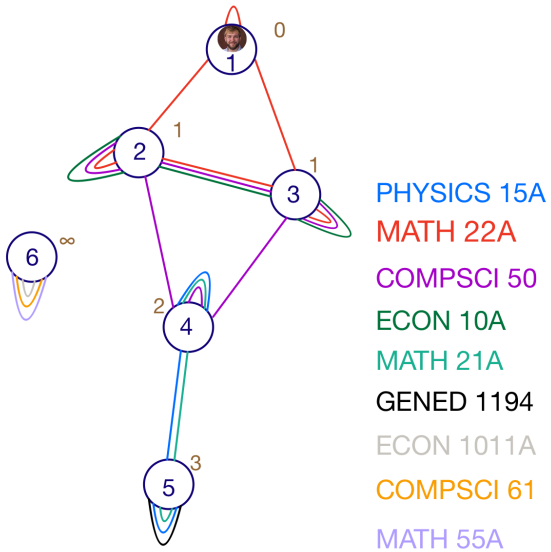


Figure 2: Graph for adjacency matrix of equation 2

Here we added the shortest distance from Dusty to each node in brown. We will show how to compute this distance (Dusty score) in a later section.

3.3 The simple graph adjacency matrix

In order to calculate the Dusty score, we notice that we are not concerned with the exact number of direct edges between any two nodes, but rather whether or not any direct edge exists between them. In other words, if we know that students i and j share at least one direct edge, whether they share 1, 2 or 3 edges does not matter. This makes intuitive sense because in calculating the shortest distance between two nodes, we are finding the shortest

walk between the nodes, which is affected by the existence of edges between any two nodes on the graph but not by the number of such edges. This means that we can reduce our original graph into a **simple graph** in order to harness results that are uniquely applicable to simple graphs.

Definition A **simple graph** is a graph containing no loops or multiple edges.

We can reduce the original graph to a simple graph by collapsing multiple direct edges between the same two nodes into a single edge and removing all self-loops. The figure below shows the corresponding simple graph of our original graph.

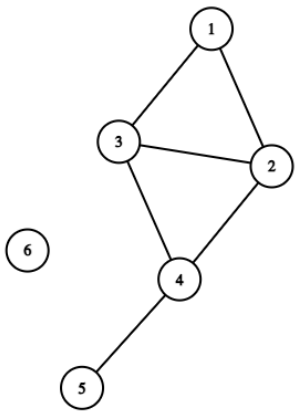


Figure 3: Graph for adjacency matrix of equation 4

From the properties of simple graphs, we can obtain a more specific way of characterizing the adjacency matrix of simple graphs.

Observation The adjacency matrix for a simple graph with n nodes is an $n \times n$ matrix where entry a_{ij} is '1' if the ith node is adjacent to the jth node and '0' otherwise.

Noting that students i and j are adjacent in the simple graph representation if and only if they share at least 1 class, and all self-loops are removed in converting the original graph to a simple graph, we arrive at the following observation:

Observation We can derive the simple graph adjacency matrix S by putting the entries of the original adjacency matrix A through the following function:

$$s_{ij} = f(a_{ij}) = \begin{cases} 1 & \text{if } i \neq j \text{ and } a_{ij} \neq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Applying this to our example yields the following adjacency matrix for our simple graph:

$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4}$$

One can verify that this adjacency matrix describes exactly the graph at Figure 3.

3.4 Solving APSP with matrix multiplication

Armed with the simple graph adjacency matrix, we can now present a few theorems that provide more information about the graph, including the shortest path lengths.

Theorem 3.2 Given an adjacency matrix A, the number of k-step walks between nodes i and j is the (i, j) entry in A^k. [1]

Proof. We proceed by induction. For the base case, we need to check if the number of 1-step walks between nodes i and j is the (i, j) entry in A. This is self-evident from the definition of the adjacency matrix for a simple graph.

For the inductive hypothesis, let us assume that the number of n-step walks between nodes i and j is the (i, j) entry in A^n.

$$A^n = \begin{bmatrix} — & a_1 & — \\ — & a_2 & — \\ & \vdots & \\ — & a_n & — \end{bmatrix}, \qquad A = \begin{bmatrix} | & | & & | \\ b_1 & b_2 & \dots & b_n \\ | & | & & | \end{bmatrix}$$

The (i, j) entry in A^{n+1} is the result of the vector multiplication of a_i and b_j. Invoking the inductive hypothesis, the kth entry of the row vector a_i is the number of n-step walks between nodes i and k. In the base case, we have also shown that the kth entry of the column vector b_j is the number of 1-step walks between nodes j and k.

Thinking of node k as an intermediary, by the multiplication principle of counting, multiplying the kth entry of a_i and the kth entry of b_j gives us the number of $(n + 1)$-step walks from node i to j passing through k as an intermediary. By the property of vector multiplication, the (i, j) entry in $A^n + 1$ is thus the sum of all such products for $1 \leq k \leq m$. Since the cases are mutually exclusive (i.e. any $(n + 1)$-step walk that passes through node k in the nth step cannot also pass through node p in the nth step, where $1 \leq k, p \leq m$ and $j \neq p$), the above sum is equal to the total number of $(n + 1)$-step walks from node i to j considering all possible intermediaries. Hence the (i, j) entry in A^n gives the number of $(n + 1)$-step walks between nodes i and j.

Since the theorem is true for $k = 1$ and the statement is true for n implies that it is also true for $n + 1$, by mathematical induction, the theorem holds for all $k \geq 1$. □

For example, here is A^2 for the example graph in Figure 3. From the theorem above, the entry a_{ij} is the number of 2-step walks from node i to j.

$$A^2 = \begin{bmatrix} 2 & 1 & 1 & 2 & 0 & 0 \\ 1 & 3 & 2 & 1 & 1 & 0 \\ 1 & 2 & 3 & 1 & 1 & 0 \\ 2 & 1 & 1 & 3 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Let's examine the first row. Note that a_{11} is 2, which are the walks $1 \to 2 \to 1$ and $1 \to 3 \to 1$. Then for $a_{12} = 1$, there is only one 2-step walk from 1 to 2 which is $1 \to 3 \to 2$. The most important value in this matrix are $a_{13} = 2$ as this shows that there exists walks from node 1 to node 3. Previously in A this entry is zero, as there do not exist any 1-step walks (direct edges) between from node 1 to node 3. As node 1 now gains the ability to travel to node 3 through 2-step walks, we can conclude that the shortest walk length from node 1 to node 3 is 2.

Observation If the (i, j) entry in A is zero, but the (i, j) entry is A^2 is non-zero, then the shortest distance between node i and j is 2.

Now let's generalize for higher powers of A. If the (i, j) entry in A^t is always zero for $t = 1, 2, \ldots, k-1$, but the (i, j) entry in A^k is non-zero, then this means that there does not exist any path between i and j of lengths 1 to $k-1$, but there exists some paths from i to j with length k. Therefore, we can similarly conclude that the shortest distance between node i and j is k.

Theorem 3.3 The shortest distance between nodes i and j is the minimum k such that the (i, j) entry in A^k is non-zero. If no such minimum k exists, then the shortest distance is defined as infinity.

Proof. Let the shortest distance between nodes i and j be k. This requires that the (i, j) entry in A^k to be non-zero, since there is at least one k-step path between i and j. Assume for contradiction that k is not the minimum power of A where the (i, j) entry of such matrix is non-zero. Since k is not the minimum, there exists $l < k$ such that the (i, j) entry of A^l is non-zero. This means there exists a path between i and j of length l. Since $l < k$ but k is the length of the shortest path between nodes i and j, we arrived at a contradiction. Therefore, k must be smallest power of A such that the (i, j) entry of A^k is non-zero. □

We now have a relationship that connects matrix multiplication to the shortest path solution. We continue by finding a function that utilizes this relationship to take an adjacency matrix as input and output the distance matrix.

Since we are only interested in the presence of paths between nodes (whether the entries are non-zeroes), and not the number of such paths, it is convenient to create a function that simplifies the matrix so that all non-zero entries are standardised to a single value.

Definition Let $f(A)$ be the function on the square matrix A that maps non-zero entries to 1 and zero entries to infinity.

One can think of this function of "booleanizing" the matrix into true (1) and false (∞). For example,

$$f(\begin{bmatrix} 3 & 0 & 4 \\ 0 & 9 & 3 \\ 1 & 4 & 2 \end{bmatrix}) = \begin{bmatrix} 1 & \infty & 1 \\ \infty & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

We can now use this function along with the previous theorem to compute the shortest path lengths. Note that the (i, j) entry on $f(A^k)$ is 1 if there exists a k-step path from i to j and infinity otherwise.

With the motivation of associating the path length with the matrix, we can **weight** this function with k, so that the (i, j) entry of $k \cdot f(A^k)$ is k if there exists a k-length path from i to j and infinity otherwise.

Definition In this paper we define the minimum of $m \times m$ matrices $A_1, A_2, \ldots, A_p$ as the matrix D such that for all $1 \leq i, j \leq m$ the (i, j) entry of D is the minimum (i, j) entry across all $A_1, A_2, \ldots, A_p$. D can be written as $\min(A_1, A_2, \ldots, A_p)$.

We need one more observation before arriving at the final APSP solution.

Observation Given a connected graph of m nodes, the shortest distance between any two nodes is less than or equal to $m - 1$.

This limit is apparent when one considers the case where all the nodes are connected like a chain, so that the two nodes at each end has $m-1$ edges between them. This is important as we only need to compute up to A^{m-1} to get all possible shortest path lengths.

Theorem 3.4 Given a $m \times m$ adjacency matrix A, the distance matrix Q can be computed by

$$Q = \min(f(A), 2 \cdot f(A^2), ..., (m-1) \cdot f(A^{m-1})$$

Or more succinctly

$$Q = \min_{i=1}^{m-1}(i \cdot f(A^i))$$

Proof. We aim to prove that for any entry q_{ij} of Q is equal to the shortest distance between the nodes i and j. Let the shortest distance between node i and j be k. Then there do not exists any paths of length 1 to $k-1$, but at least one path of length k. Therefore, the (i, j) entries of $1 \cdot f(A), 2 \cdot f(A^2), \ldots, (k-1) \cdot f(A^{k-1})$ are all infinity, and the (i, j) entry for $k \cdot f(A^k)$ is k. Notice that for any $p > k$, the (i, j) entry for $p \cdot f(A^p)$ is either p or infinity.

Therefore, the minimum value of (i, j) across all these matrices is k.

$$\begin{aligned} q_{ij} &= \min(f(A)_{ij}, 2 \cdot f(A^2)_{ij}, \cdots, k \cdot f(A^k)_{ij}, \cdots, p \cdot f(A^p)_{ij}) \\ &= \min(f(A)_{ij}, 2 \cdot f(A^2)_{ij}, \cdots, k \cdot f(A^k)_{ij}) \\ &= k \cdot f(A^k)_{ij} \\ &= k \end{aligned}$$

From the observation on the upper bound of $k = m - 1$, we can generalize to all entries of Q by including up to the $m - 1$ term.

$$q_{ij} = \min(f(A)_{ij}, 2 \cdot f(A^2)_{ij}, \cdots, (m-1) \cdot f(A^{m-1})_{ij})$$

Then, by using our definition of the minimum of matrices, we can represent this as

$$Q = \min\left(f(A), 2 \cdot f(A^2), ..., (m-1) \cdot f(A^{m-1}) = \min_{i=1}^{m-1} (i \cdot f(A^i))\right.$$

□

Applying this procedure to our example gives the following distance matrix.

$$Q = \begin{bmatrix} 0 & 1 & 1 & 2 & 3 & \infty \\ 1 & 0 & 1 & 1 & 2 & \infty \\ 1 & 1 & 0 & 1 & 2 & \infty \\ 2 & 1 & 1 & 0 & 1 & \infty \\ 3 & 2 & 2 & 1 & 0 & \infty \\ \infty & \infty & \infty & \infty & \infty & 0 \end{bmatrix}$$

The Dusty score can be read from the first row, which is

$$\begin{bmatrix} 0 & 1 & 1 & 2 & 3 & \infty \end{bmatrix}$$

This agrees exactly with the graph at Figure 2.

4 Computational challenges

Now that we have a procedure for calculating the Dusty score of a given student population, we can apply this to the Harvard Class of 2026. However, as will be shown in Section 4.1, this process will take a hundred days for a student body of 2000. Thus, we want to explore if there are results in linear algebra that can shorten the computational process.

4.1 Number of operations

We now analyze the number of operations required to compute the distance matrix. Here we use the big-O notation (insert citation) to investigate the growth of the number of operations with the size of the matrix.

Observation Given an $n \times n$ matrix A, the number of operations required to compute $A \times A$ is $O(n^3)$.

It then follows that A^k takes $O(n^3 k)$ operations.

Observation Given a graph with n vertices, the number of operations required to compute the distance matrix Q naively using the procedure in Theorem 3.4 is $O(n^5)$.

This is a horrible time efficiency. For 2,000 students, this requires number of operations at the magnitude of 10^{15}, which requires 100 days to compute on a modern computer! [7]

4.2 Spectral Theorem to the Rescue

Fortunately, since our graph is simple, unweighted and undirected, and the corresponding adjacency matrix is symmetric, we can use the spectral theorem to speed up the procedure.

Theorem 4.1 A $n \times n$ real matrix A is orthogonally diagonalizable if and only if A is symmetric. This means there exists a diagonal matrix D and orthonormal matrix S such that $A = SDS^T$.

The proof can be found in [5].

Theorem 4.2 Given a symmetric matrix A, there exists diagonal matrix D and orthonormal matrix S such that $A^k = SD^kS^T$.

Proof. Using the spectral theorem we can expand A^k into

$$A^k = \underbrace{SDS^T \cdots SDS^T}_{k \text{ times}}$$

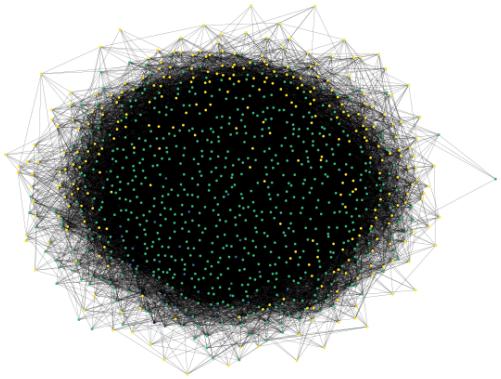
Since matrix multiplication is associative, we can evaluate S^TS which occurs with the S^T from a preceding term and S from the next term. Note that since S is orthonormal, $S^TS = I$. Therefore, $A^k = SD^kS^T$ as required. □

Now we can use the spectral theorem to speed up our computation of the distance matrix Q, since the adjacency matrix A is symmetric and $Q = A^{n-1} = SD^{n-1}S^T$. The orthonormal matrix S can be found through the Gram-Schmidt algortihm which requires $O(n^3)$ time [4]. The transpose only needs $O(n^2)$ reassignments. The computation of D^{n-1} is $O(n^3)$ from a previous observation. The matrix multiplication of the three terms is $O(n^3)$. Therefore, the overall time complexity for the computation of Q is $O(n^3)$.

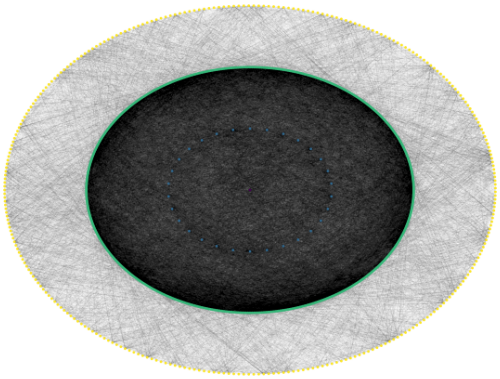
For completeness, we also like to highlight that the conventional APSP solution through min-plus matrix multiplication also gives a time complexity of $O(n^3)$. With certain optimizations this can be pushed to $O(n^{2.38})$ [2]

5 Calculating the Dusty Score

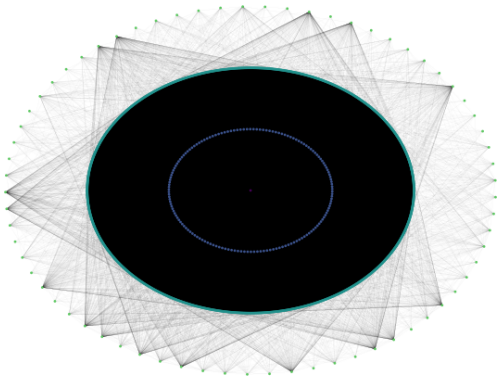
We applied our procedure to the class enrollment data provided by the registrar. We also simulated the enrollment of 1,000 and 7,095 students weighted only by the class sizes through the Q guide (without the correlations like that of CS50 and MATH 22A). The code for the web scraping, simulation and Dusty score calculation can be found here.



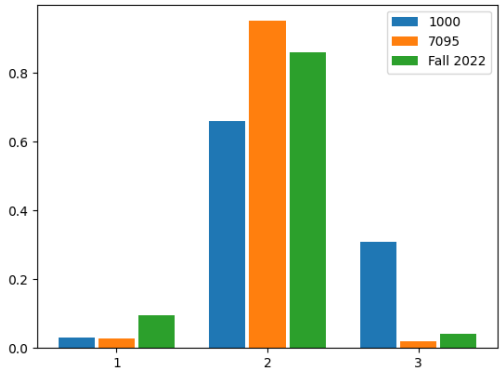
(a) Network of 1,000 students simulated from Fall 2021 class enrollment data



(b) Network of 1,000 students arranged in concentric shells of Dusty scores



(c) Network of Fall 2022 first-years arranged in concentric shells of Dusty scores



(d) The distribution of Dusty scores

We end with the observation that the maximum Dusty score is 3 in both the 1,000 case, the 7,095 case and the actual case for Fall 2022 first-years.

Observation All Harvard first-years in the Fall of 2022 either take Math 22 (10%), have a classmate who takes Math 22 (86%), or share a mutual classmate with a Math 22 student (4%). The maximum Dusty score is 3.

References

[1] James Baglama. "Using Linear Algebra to Solve Graph Problems". In: (Nov. 2017). URL: https://www.math.uri.edu/~jbaglama/classes/2017-2018/fall/mth215/project2.pdf.

[2] Don Coppersmith and Shmuel Winograd. "Matrix multiplication via arithmetic progressions". In: *Proceedings of the nineteenth annual ACM symposium on Theory of computing*. 1987, pp. 1–6.

[3] Jeremy T. Fineman. *Graph Algorithms in the Language of Linear Algebra*. Society for Industrial and Applied Mathe, 2011, pp. 45–58.

[4] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. The John Hopkins University Press, 1996, pp. 230–233.

[5] Dusty Grundmeier. *Spectral Theorem*. 2022.

[6] Antti Laaksonen. *Competitive Programmer's Handbook*. July 2018, pp. 109–112.

[7] *Microprocessor Hall of Fame*. 2008. URL: https://web.archive.org/web/20080406154333/http://www.intel.com/museum/online/hist_micro/hof/.

[8] Wikipedia. *Shortest path problem*. 2022. URL: https://en.wikipedia.org/wiki/Shortest_path_problem.