QUIZ 4

purdue university · cs 51500

David Gleich

matrix computations

June 27, 2023

Purdue Honor Code

The purpose of the Purdue University academic community is to search for truth and to endeavor to communicate with each other. Self-discipline and a sense of social obligation within each individual are necessary for the fulfillment of these goals. It is the responsibility of all Purdue students to live by this code, not out of fear of the consequences of its violation, but out of personal self-respect. As human beings we are obliged to conduct ourselves with high integrity. As members of the civil community we have to conduct ourselves as responsible citizens in accordance with the rules and regulations governing all residents of the state of Indiana and of the local community. As members of the Purdue University community, we have the responsibility to observe all University regulations.

To foster a climate of trust and high standards of academic achievement, Purdue University is committed to cultivating academic integrity and expects students to exhibit the highest standards of honor in their scholastic endeavors. Academic integrity is essential to the success of Purdue University's mission. As members of the academic community, our foremost interest is toward achieving noble educational goals and our foremost responsibility is to ensure that academic honesty prevails.

Exam Rules

AP_____ This quiz is a *open to all class and internet resources* except talking with people.

AP_____ The contents of this quiz are protected by copyright. Posting any piece of this quiz – including excepts that may be subject to fair-use rights – to any database, online resource, electronic medium, or other similar repository will be considered academic dishonesty (see below). Exams may be individually watermarked to identify violations.

AP_____ You may not consult with anyone other than the professor and the TA.

AP_____ Any behavior consistent with academic dishonesty (i.e. cheating) *will not be tolerated and may result in a 0 even a failing grade in the class*.

AP You have until the Gradescope or Blackboard submission closes to com plete and submit the exam. No exceptions.

Write your name, PUID, and sign below to indicate you agree with the statement:

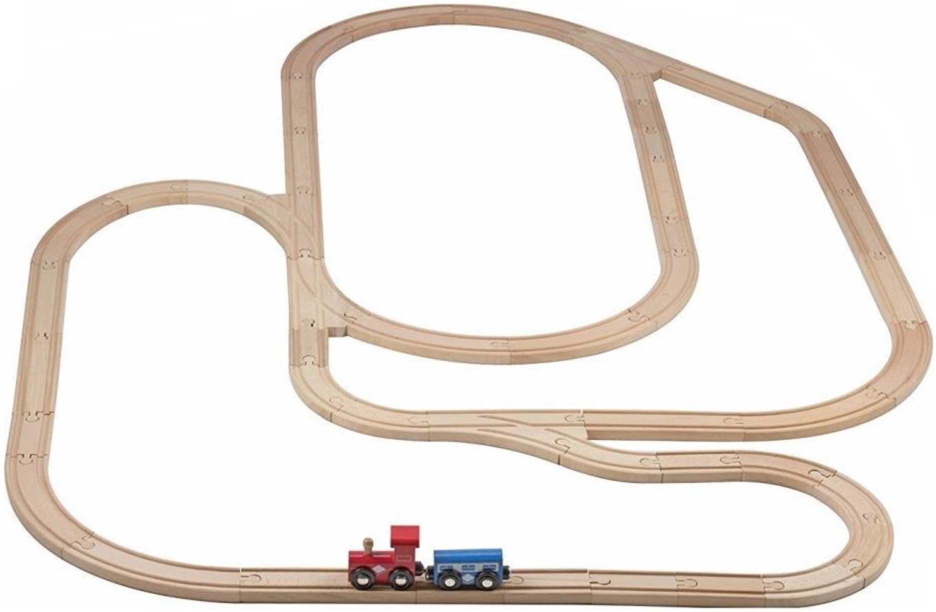
The remainder of this exam represents my own work.

(Your Name) Aditya Patel (PUID) 0035573514

Aditya Patel
(Signature)

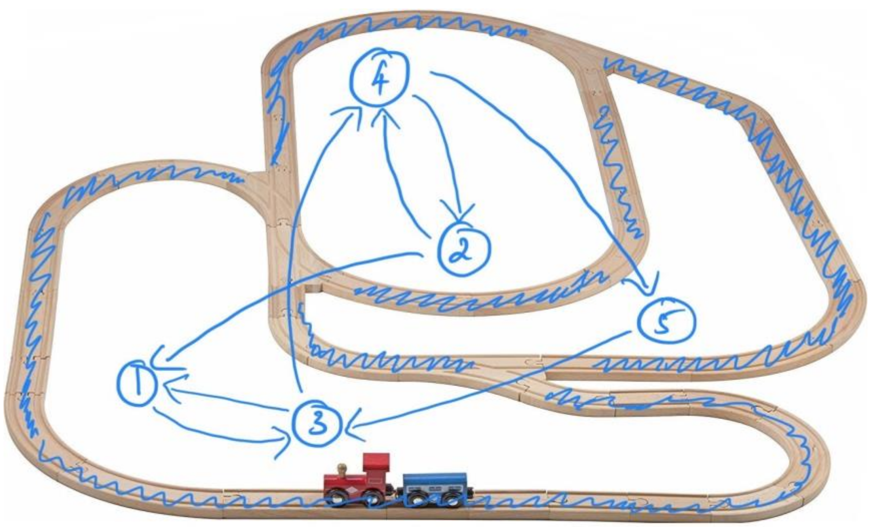
Problem 1 *(30 points)*

In class we saw random walks and random surfers. Here, we are going to investigate random play. A common household toy in the US is a wooden train track. An example from a Walmart advertisement[1] is



Imagine a child playing with this train set where the train starts as indicated in the picture and moves to the left (i.e. in the direction of the red car). At any intersection where there is a choice, the child will randomly pick a direction. Note that an intersection may not offer a choice based on which way the train is going. We are going to use a graph to build a model to determine how often each section of track between intersections is visited under the child's random play model.

A random walk on the graph in following picture illustrates a simple model of what happens.



[1] https://www.walmart.com/ip/Trains-All-PCS-Railway-Set-Is-With-Toys-Sets-52-Thomas-Tracks-For-Bonus-Compatible-Kids-Systems-2-Wooden-Major-Car-By-Play22-Brands-Train-Original-Toy/85696313

(10 points) Use that graph model to find the long term fraction of time that the train will spend on each segment 1-5 based on what you learned in class. Justify any entries of 0 based on the strong component structure.

The long-term fractions of time can be determined by using the following code in Julia.

```
using LinearAlgebra
A = [0 0 1 0 0
    1 0 0 1 0
    1 0 0 1 0
    0 1 0 0 1
    0 0 1 0 0]

## Get the inverse of the diagonal degree matrix
d = vec(sum(A, dims=2))
D_inv = Diagonal(1.0 ./ d)

## Find P
P = Matrix(A' * D_inv)
vals, vecs = eigen(P)
## Find the eigenvector corresponding to the largest eigenvalue
max_val_ind = findmax(abs.(vals))[2]
println("Largest Eigenvector:")
probs = Float64.(vecs[:, max_val_ind]/sum(vecs[:, max_val_ind]))
```

This results in the following results corresponding to each node:

```
0.22222222222222252
0.11111111111111088
0.3333333333333332
0.2222222222222225
0.11111111111111092
```

(10 points) Now suppose that the train starts in the opposite direction. Give the new graph model and show the long term probability. Justify any entries of 0 based on the strong component structure.

Taking the transpose of A in the Julia code above to turn the in-edges into out-edges (reversing the graph directions) and then running the code once more, we see the following results corresponding to each node.

```
0.11111111111111116
0.22222222222222215
0.2222222222222222
0.3333333333333335
```

```
0.11111111111111113
```

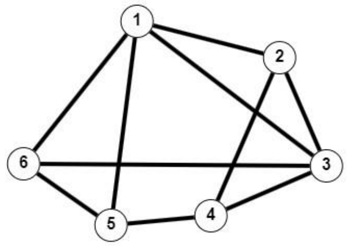
(10 points) Now imagine we have two children playing. One of whom wants to have the train go, and the other who keeps picking the train up off the train and putting it back randomly. Let's approximate the behavior of what happens with a PageRank model. Suppose that 90% of the time, the train-child is allowed to move the train in a fixed direction. The other child will come with probability 1/8 and move the train somewhere new with a random direction. Use a PageRank model with α = 0.875 to find the long-term fraction of time on each segment of track going each of the two directions.

Using the simple pagerank algorithm with alpha = 0.875, we observe the following results:

```
0.21852551984877125
0.12060491493383743
0.32173913043478264
0.21852551984877125
0.12060491493383743
```

Problem 2 *(20 points)*

Consider this graph



(10 points) Give the Laplacian matrix for this graph.

```
using LinearAlgebra
using MatrixNetworks
using SparseArrays

A = [0 1 1 0 1 1;
    1 0 1 1 0 0;
    1 1 0 1 0 1;
    0 1 1 0 1 0;
    1 0 0 1 0 1;
    1 0 1 0 1 0;]

D = Diagonal(vec(sum(A, dims=2)))
L = D – A
```

Which gives the following Laplacian Matrix:

```
 4 -1 -1  0 -1 -1
-1  3 -1 -1  0  0
-1 -1  4 -1  0 -1
 0 -1 -1  3 -1  0
-1  0  0 -1  3 -1
-1  0 -1  0 -1  3
```

(5 points) Give the incidence matrix for this graph.
To find the incidence matrix B, we can use this Julia code:

```
A = [0 1 1 0 1 1
    1 0 1 1 0 0
    1 1 0 1 0 1
    0 1 1 0 1 0
    1 0 0 1 0 1
    1 0 1 0 1 0]

# Get the number of vertices
num_vertices = size(A, 1)

# Find all edges in the graph
edges = []
for i in 1:num_vertices
   for j in i+1:num_vertices
      if A[i, j] == 1
         push!(edges, (i, j))
      end
   end
end

# Get the number of edges
num_edges = length(edges)

# Initialize the incidence matrix B
B = zeros(Int, num_edges, num_vertices)

# Fill the incidence matrix B
for (k, (i, j)) in enumerate(edges)
   B[k, i] = 1
   B[k, j] = -1
end

# Print the incidence matrix B
print(B)
```

Which results in the following B matrix

```
1 -1 0 0 0 0
1 0 -1 0 0 0
1 0 0 0 -1 0
1 0 0 0 0 -1
0 1 -1 0 0 0
0 1 0 -1 0 0
0 0 1 -1 0 0
0 0 1 0 0 -1
0 0 0 1 -1 0
0 0 0 0 1 -1
```

(5 points) Give the Fiedler vector (the eigenvector used for spectral clustering) for this graph with the first entry x_1 positive.

To find the Fiedler vector, we can use this Julia code:

```
using LinearAlgebra
using MatrixNetworks
using SparseArrays

A = [0 1 1 0 1 1;
     1 0 1 1 0 0;
     1 1 0 1 0 1;
     0 1 1 0 1 0;
     1 0 0 1 0 1;
     1 0 1 0 1 0;]

f2 = fiedler_vector(sparse(A))[1]
f2 .*= sign(f2[1])
```

The fiedler vector f2 is then

```
6-element Vector{Float64}:
  0.11619967077072714
 -0.304407036570377
 -0.1161996707707274
 -0.23664291968450613
  0.23664291968450415
  0.30440703657037743
```