Technische Universiteit
Eindhoven
University of Technology

Department of Mathematics and Computer Science

Homework II

2MMD30 Graph and algorithms

Benedetto Buratti & Magnus Malmström

Eindhoven, Mars 2017

Chapter 1

Exercises

Here will all the exercises for the homework be stated. The structure is that we first present the problem and after that our solution to the problem

1.1 Exercise one

The knights graph is the graph on 64 vertices with a vertex for every square of a chess-board and two vertices being adjacent if they are a knights move away from each other (formally, we have vertices $v_{i,j}$ for$1 \leq i, j \leq 8$ and an edge $(v_{i,j}, v_{i',j'})$ if either $|i-i'| = 1$ and $|j-j'| = 2$ or $|i-i'| = 2$ and $|j-j'| = 1$). Show that the knights graph has treewidth at most 16

1.1.1 Solution

To solve this question we will consider the "cops and robber game" and show that there is a strategy so that 17 cops can win the game. Consider every square on the chessboard as a vertex and label the vertices $v_{i,j}$ where $1 \leq i, j \leq 8$ so that vertex $v_{1,1}$ is the square $(A, 1)$, $v_{1,2}$ $(A, 2)$, $v_{2,1}$ $(B, 1)$ and so on.

The strategy is that that you start placing one cop at $v_{1,1}$ then the next one at $v_{2,1}$ and so on until the fist row is filled with cops. Then you move on the next row and fill it with cops as well so we now have 16 cops on the board. Then add the 17^{th} cop at vertex $v_{1,3}$. After that we move the cop from $v_{1,1}$ to $v_{2,3}$, the cop at $v_{2,1}$ to $v_{3,3}$ and so on until the third row is full. Then you move the cop from $v_{8,1}$ to $v_{1,4}$ and to the same thing for the second row. A general formula for how the cop will move is:

$$v_{i,j} = \begin{cases} v_{i+1,j} \text{ if } i \leq 8 \\ v_{1,j+1} \text{ ,} i = 8 \\ \text{Stop when the robber has been caught} \end{cases}$$

This strategy works since the cops builds a "wall" across the chess board which the robber cant pass through. Why we need 17 cops and not only 16 is that robber can go passed one line of cops since the knight can jump two in some direction. This means we always need to have two full rows.

Since the "cops and robber game" can be won with $16 + 1$ cops on a knight graph we have now showed that a knight graph at most has threewith 16. □

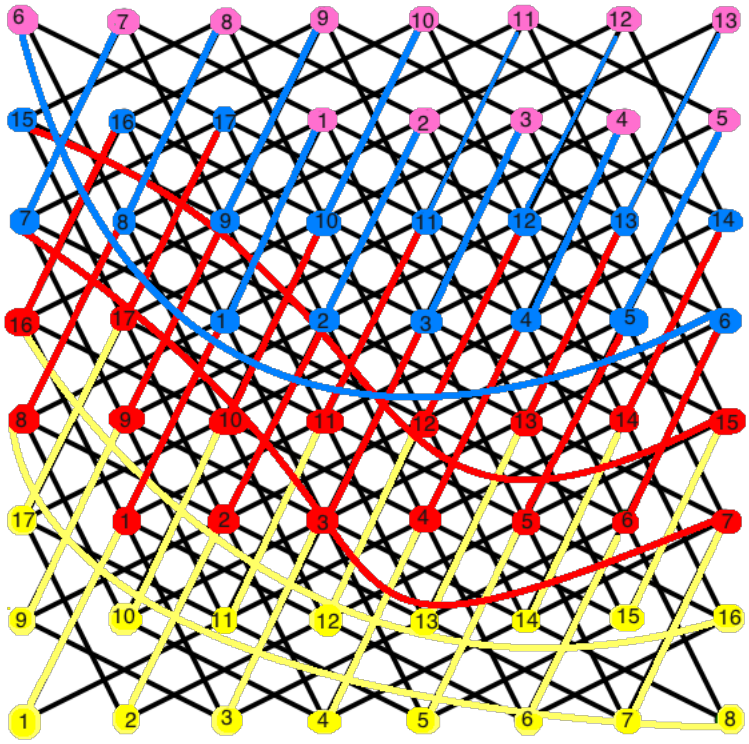

Figure 1.1: Picture of how the cops win the cop and robber game. The cops change color when they move, string from the botom to the top.

1.2 Exercise two

Consider an unweighted graph G which has a minimum cut of value k. For a $\alpha \geq 1$, a cut S is called a $\alpha-$min cut if its value is at most αk. Show that the number of $\alpha-$min cuts in any graph is at most $n^{2\alpha}$.

1.2.1 Solution

From the definition of $\alpha-$ min cut S has at most αk (we also assume α is an integer) cuts which gives us that

$$P[S \text{ is hit in round } 1] \leq \frac{\alpha k}{nk/2} = \frac{2\alpha}{n}$$

and

$$P[S \text{ is hit in round} i+1 | S \text{ survives round} 1, ..., i] \leq \frac{\alpha k}{(n-i)k/2} = \frac{2\alpha}{n-i}$$

so that

$$\begin{aligned} P[S \text{ survives until } 2\alpha \text{ vertices remain }] \geq & (1 - \frac{2}{n}) \times (1 - \frac{2}{n-1}) \times ... \times (1 - \frac{2}{2\alpha}) \\ = & (\frac{n-2}{n}) \times (\frac{n-3}{n-1}) \times \frac{(n-4)}{(n-2)} \times ... \times (\frac{2\alpha-1}{2\alpha+1}) \times (\frac{2\alpha-2}{2\alpha}) \\ = & \frac{(2\alpha-1)(2\alpha-2)}{n(n-1)} \geq 1/\binom{n}{2\alpha}. \end{aligned}$$

We now difine K to be random min-cut prosses until 2α vertices remains, and that pick a random cut in the remaining graph. Then we can see that

$$\begin{aligned} P[S \text{ survives } K] &= P[S \text{ survives until } 2\alpha \text{ vertices remains}] \times P[S \text{ survives random cut}] \\ &\geq \frac{1}{\binom{n}{2\alpha}} \times \frac{1}{2^{2\alpha-1}} \geq \frac{(2\alpha)!}{2^{2\alpha}} \times \frac{1}{n^{2\alpha}} \\ &\geq \frac{1}{n^{2\alpha}}. \end{aligned}$$

Since the sum of all probabilities is 1 this tells us that it at most can exist $n^{2\alpha}$ $\alpha - \min$ cuts in a graph. □

1.3 Exercise three

Given n distinct real numbers. Suppose we want to find the maximum number among them. A simple algorithm is to scan the numbers one by one and store a *temporary max* which remembers the index of the largest number seen so far. While scanning the i^{th} number, if it is larger than *temporary max*, we update the value of *temporary max* to be i; else we keep *temporary max* the same. When we have scanned all the elements, *temporary max* will give the index of the largest number. Now suppose that we first randomly order the n input numbers. What is the expected number of times we will update the value of temporary max in the above algorithm?

1.3.1 Solution

First of all let's define a random variable X to indicate the number of updates of our variable.

Considering the previous algorithm, we can calculate the expected number of updates, $E[x]$, by using backward analysis; indeed we start our reasoning from the n-step of our algorithm.We note that the expected number of updates for the last step is recursively related with the penultimate step result. Indicating with A our array, with $A[n]$ the array's n-th element and with $f(n)$ the number of updates at the n-th step, we have two different situations:

- if **A[n]** max**:** $f(n) = f(n-1) + 1$; which has probability $Pr[A[n] == max] = \frac{1}{n}$
- if **A[n]** $\neg max$**:** $f(n) = f(n-1)$; which has probability $Pr[A[n]! = max] = \frac{n-1}{n}$

At this point we can derive a recursive formula to compute the number of updates for the n-th step:

$$f(n) = \frac{f(n-1)+1}{n} + f(n-1)\frac{n-1}{n} = f(n-1) + \frac{1}{n} \tag{1.1}$$

We can use this general formula to compute any step of our algorithm, as showed by the following enumeration:

- $f(0) = 0$
- $f(1) = 1$
- $f(2) = 1 + \frac{1}{2}$
- $f(3) = 1 + \frac{1}{2} + \frac{1}{3}$
- ...

From the previous argument we can deduce that the expected number of updates $E[X]$ for a n-dimensional array, is nothing else that an harmonic number H_n.

$$E[X] = H_n = \sum_{n=0}^{\infty} \frac{1}{n} \tag{1.2}$$

So we can compute $E[X]$ by calculating H_n, but since there is no closed form to compute an harmonic number, we are going to exploit the fact that an harmonic series can be approximated by a natural logarithm, since the $\int_0^{\infty} \frac{1}{x} = ln(x)$.

However we can be even more precise by using this well-known asymptotic expansion, for harmonic numbers:

$$H_n = \ln n + \gamma + \frac{1}{2n} - \sum_{k=1}^{\infty} \frac{B_{2k}}{2kn^{2k}} = \ln n + \gamma + \frac{1}{2n} - \frac{1}{12n^2} + \frac{1}{120n^4} - \cdots \tag{1.3}$$

Where $\frac{B_{2k}}{2kn^{2k}}$ are Bernoulli numbers and γ is the *Euler-Mascheroni constant*, defined as following:

$$\lim_{n \to +\infty} (H_n - \ln(n)) = \gamma \tag{1.4}$$

which has an approximate value of $\gamma \approx 0.5772156649$.

1.4 Exercise four

Recall that the normalised Laplacian matrix of a $d-$regular graph G given by L_G has its maximum eigenvalue λ_n given by

$$\lambda_n = \max_{x \in \mathbb{R}^n} \frac{\sum_{(i,j) \in E} (x_i - x_j)^2}{d \sum_i x_i^2}.$$

A max-cut in a graph is a cut which has the maximum number of edges crossing it. Let M be the value of the max-cut in the graph and $|E|$ the total number of edges in the graph. Prove that

$$\frac{M}{|E|} \leq \frac{\lambda_n}{2}.$$

1.4.1 Solution

Split thar graph in three peaces X, Y and $\overline{S}$ such that the max-cut M is between X and Y and $\overline{S}$ is disconected parts of the graph. Notice that even if bouth X and Y are connected to $\overline{S}$ there is no edge that connect X and Y throw $\overline{S}$.

We can now see that the max-cut $M = E[X, Y]$ and that $|E| = d|S|/2$ which gives us that:

$$2d\frac{M}{|E|} = 2d\frac{E[X,Y]}{d|S|/2} = 4\frac{E[X,Y]}{|S|}.$$

Now lets study λ_n and choose a y such that it is -1 in X 1 in Y and 0 in $\overline{S}$. Since λ_n is a max it must be smaller or equal to it with y.

$$\lambda_n = \max_{x \in \mathbb{R}^n} \frac{\sum_{(i,j) \in E} (x_i - x_j)^2}{d \sum_i x_i^2} \geq \frac{\sum_{(i,j) \in E} (y_i - y_j)^2}{d \sum_i y_i^2}$$

We can also see that

$$\sum_{(i,j) \in E} (y_i - y_j)^2 = \sum_{(i,j) \in E[X,Y]} 2^2 + \sum_{(i,j) \in E[X,\overline{S}]} 1^2 + \sum_{(i,j) \in E[X]} 0^2 + \sum_{(i,j) \in E[Y,\overline{S}]} 1^2 + \sum_{(i,j) \in E[Y]} 0^2 =$$
$$4E[X,Y] + E[X,\overline{S}] + E[Y,\overline{S}]$$

and

$$\sum_i y_i^2 = \sum_{i \in E[\overline{S}]} 0^2 + \sum_{i \in E[X]} 1^2 + \sum_{i \in E[Y]} 1^2 = |X| + |Y| = |S|.$$

This gives us that:

$$\lambda_n \geq 4\frac{4E[X,Y] + E[X,\overline{S}] + E[Y,\overline{S}]}{|S|} \geq 4\frac{4E[X,Y]}{|S|} = 2d\frac{M}{|E|}$$

□