Homework #8
Algorithms I
600.463
Spring 2017

Due on: Thursday, April 27th, 11:59pm
Late submissions: will NOT be accepted
Format: Please start each problem on a new page.
Where to submit: On Gradescope, under HW8.
Please type your answers; handwritten assignments will not be accepted.
To get full credit, your answers must be explained clearly,
with enough details and rigorous proofs.

May 13, 2017

1 Problem 1 (20 points)

A bipartite graph $G = (V, E)$, where $V = L \cup R$, is *d-regular* if every vertex $v \in V$ has degree exactly d.
Ref. to Online

(a) Show that for every *d-regular* bipartite graph, $|L| = |R|$.
Solution: Every edge in the graph connects a node in L and a node in R. Therefore the total number of edges can be expressed as both $|L|d$ and $|R|d$, which implies $|L| = |R|$.

(b) Model the maximum *d-regular* bipartite matching as a max-flow problem. Show that the max-flow value from s to t in the formulation is $|L|$.
Solution:
Reduce the bipartite matching problem to a network flow problem, as in figure 26.3 in CLRS. Every edge has unit capacity. Consider the following flow function: every edge out of s or into t has unit flow, and all the other edges have flow $1/d$. This is a valid flow, and is maximum since every edge from s is saturated. Therefore, the max-flow value is $|L|$.

(c) Prove that every *d-regular* bipartite graph has a matching of cardinality $|L|$.
Solution:
Since every edge in the max-flow formulation has integral capacity, the integrality theorem (Theorem 26.11) guarantees that there is an integral flow with value $|L|$. Such integral flow corresponds to a matching of cardinality $|L|$.

2 Problem 2 (20 points)

In the *maximum k-cut* problem, we are given an undirected graph $G = (V, E)$, and non-negative weights $w_{ij} \geq 0$ for all $(i, j) \in E$. The goal is to partition the vertex set V into k parts $V_1, \ldots, V_k$ so as to maximize the weight of all edges whose endpoints are in different parts (i.e., $\max_{(i,j)\in E: i\in V_a, j\in V_b, a\neq b} w_{ij}$).
Give a randomized $\frac{k-1}{k}$-approximation algorithm for the MAX k-CUT problem.
Hint: please review Chapter 5.1 of the book—"The Design of Approximation Algorithms" (Williamson and Shmoys 2010), and try to solve this problem using similar ideas.
Answer:
Let's consider an algorithm as the following:

- Set k vertex parts: $V_1 \leftarrow \emptyset, V_2 \leftarrow \emptyset, \ldots, V_k \leftarrow \emptyset$
- For $i \leftarrow 1$ to $|V|$: Obtain a number x uniformly at random from $[0, 1]$. If $j/k \leq x < (j + 1)/k$ for some j, place $\{i\}$ into V_{j+1}.

This algorithm actually uniformly picks a bin to put a vertex and this procedure is clearly running in polynomial time.

Claim 2.1. *The algorithm is a $\frac{k-1}{k}$-approximation algorithm.*

Proof. Let X_{ij} be the random variable:
$X_{ij} = \begin{cases} 1 & \text{if for some } a, b, i \in V_a, j \in V_b, a \neq b \\ 0 & \text{otherwise} \end{cases}$ and let $W = \sum_{(i,j)\in E} w_{ij} X_{ij}$.
Now, let's consider the expectation of W and denote OPT as the optimal weight

of the problem.

$$\begin{aligned}
E[W] &= \sum_{(i,j)\in E} w_{ij} Pr[i \in V_a, j \in V_b \text{ for some } a \neq b] \\
&= \sum_{(i,j)\in E} w_{ij}(1 - Pr[i \in V_a, j \in V_a \text{ for some } a]) \\
&= \sum_{(i,j)\in E} w_{ij}(1 - \sum_{a=1}^{k} Pr[i \in V_a, j \in V_a]) \\
&= \sum_{(i,j)\in E} w_{ij}(1 - \sum_{a=1}^{k} \frac{1}{k^2}) \\
&= \sum_{(i,j)\in E} w_{ij}(1 - \frac{1}{k}) \\
&= \frac{k-1}{k} \sum_{(i,j)\in E} w_{ij} \\
&\geq \frac{k-1}{k} OPT
\end{aligned}$$

$\square$