Problem Set 5 for CS 170

Problem 0. [Any questions?] (5 points)

What's the one thing you'd most like to see explained better in lecture or discussion sections? A one-line answer would be appreciated.

(Sometimes we botch the description of some concept, leaving people confused. Sometimes we omit things people would like to hear about. Sometimes the book is very confusing on some point. Here's your chance to tell us what those things were.)

Problem 1. [MSTs for directed graphs] (20 points)

Given a weighted undirected graph, G, the MST algorithms construct a subgraph of minimal weight that is connected. Can these algorithms be adapted to directed graphs? In other words, either: (1) argue informally that Prim's and Kruskal's algorithm will always construct a strongly connected subgraph of minimal weight, given any weighted strongly connected directed graph, or (2) produce a counterexample.

Problem 2. [2-Universal Hash Functions] (25 points)

Let $[m]$ denote the set $\{0, 1, \ldots, m-1\}$. For each of the following families of hash functions, say whether it is 2-universal or not, and then justify your answer:

(a) $H_a = \{h(x_1, x_2) = a_1 x_1 + a_2 x_2 \bmod m \mid a_1, a_2 \in [m]\}$, where m is some fixed prime. (Note that each $h \in H_a$ has signature $h : [m]^2 \to [m]$, i.e., it maps a pair of integers in $[m]$ to a single integer in $[m]$.)

(b) $H_b = \{h(x_1, x_2) = a_1 x_1 + a_2 x_2 \bmod m \mid a_1, a_2 \in [m]\}$, where $m = 2^k$ is some fixed power of two.

(c) $H'_n = \{h(x) \bmod m \mid h \text{ is a degree } n \text{ polynomial}\}$, where m is some fixed prime.
For which n is H'_n 2-universal?

(d) $H_c = \{\texttt{string-hash}_a() \mid a \in [1013]\}$ (see Figure 1). The family is indexed by a.

$\texttt{string-hash}_a(s)$:
Set $x := 0$.
For $i := 0$ to $\texttt{length}(s) - 1$, do:
    Set $x := a * x + s[i] \bmod 1013$.
Return x.

Figure 1: Generic hash function for strings.

Problem 3. [Collisions] (25 points)

Let H be a family of 2-universal hash functions whose output is the set $\{1, 2, \ldots, m\}$. Suppose we hash m different inputs, and let N_i denote the number of inputs that hash to i. Prove that $\mathbb{E}\left[N_1^2 + \ldots + N_m^2\right] \leq 10m$. (Here $\mathbb{E}[X]$ denotes the expected value of X.)

Problem 4. [Average-case analysis] (30 points)

(a) Suppose each element of the array $A[1..n]$ is a real number chosen uniformly and independently from the interval $[0, 1]$. Fix i. What is the probability (in terms of n and i) that $A[i]$ is the smallest element among all of $A[1..i]$? You do not need to justify your answer.

(b) The algorithm `min1()` in Figure 2 computes the minimum of an array of real numbers. Suppose the array elements are distributed as in part (a), and let X denote the number of times that line 4 is executed. Show informally that $\mathbb{E}[X] = O(\lg n)$.

HINT: You may use the following fact without further justification: $1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{k} = \log k + O(1)$.

(c) Now consider the algorithm `min2()` from Figure 2, which computes the second smallest element in an array of real numbers. Assume that the array elements are distributed as in (a). Let T denote the number of comparisons executed by this algorithm. Show informally that $\mathbb{E}[T] = n + O(\lg n)$.

(d) Suppose we apply `min2()` to completely arbitrary inputs (where no assumptions are made on the array $A[1..n]$). Does it execute in $n + O(\lg n)$ worst-case running time? Explain.

```
min1(A[1..n]):
1: Set t := A[1].
2: For i := 2 to n, do:
3:     If A[i] < t, then:
4:         Set t := A[i].
5: Return t.
```

```
min2(A[1..n]):
1: Set t := A[1] and u := ∞.
2: For i := 2 to n, do:
3:     If A[i] < u, then:
4:         If A[i] < t, then
5:             Set u := t, and then set t := A[i].
6:         else
7:             Set u := A[i].
8: Return u.
```

Figure 2: Algorithms for computing the smallest and second-smallest element of an array.

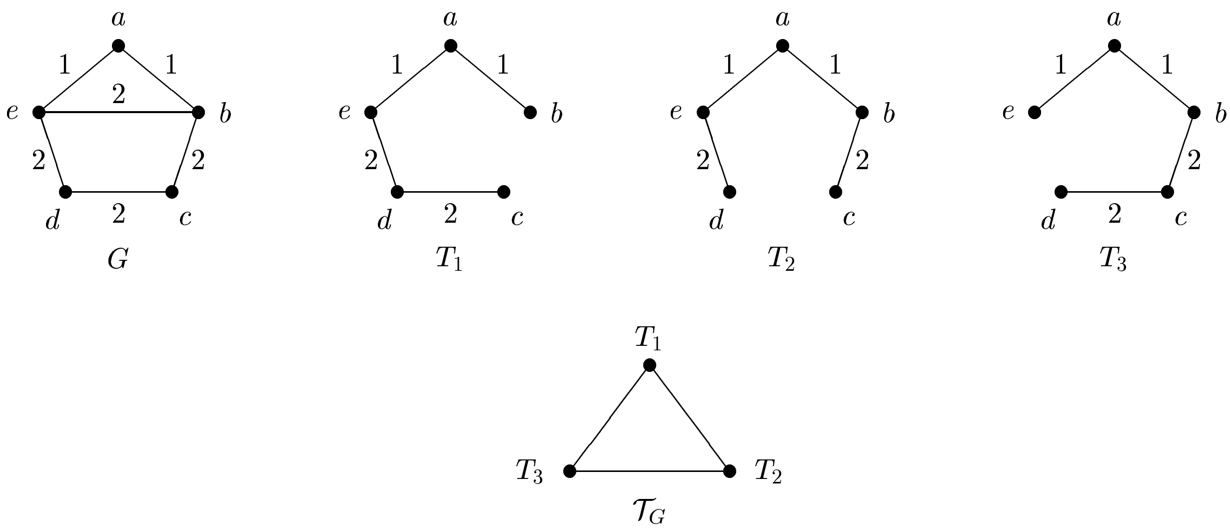

Figure 3: Example graph for the Bonus Problem.

Bonus Problem. [Minimum spanning trees] (0 points)

Let G be a connected undirected weighted graph. Then G may have many minimum spanning trees. Let $\mathcal{T}$ be the set of all minimum spanning trees of G. Let $\mathcal{T}_G = (V, E)$ be the graph with $V = \mathcal{T}$ and edges $(T_1, T_2) \in E$ (where $T_1, T_2 \in V$ are minimum spanning trees of G) if and only if there exist edges $e_1 \in T_1$ and $e_2 \in T_2$ such that deleting e_1 from T_1 and adding edge e_2 produces T_2. Observe that if $(T_1, T_2) \in E$, then so is (T_2, T_1), so we can think of $\mathcal{T}_G$ as an undirected graph. Prove that $\mathcal{T}_G$ is connected.

For example, the graph G shown in Figure 3 has three minimum spanning trees, T_1, T_2, and T_3, so $\mathcal{T} = \{T_1, T_2, T_3\}$. The spanning tree T_1 can be transformed into T_2 by deleting the edge (c, d) and adding the edge (b, c), so T_1 and T_2 are adjacent in $\mathcal{T}_G$. Similarly, we can transform T_2 into T_3 by deleting edge (d, e) and adding edge (c, d), so $\mathcal{T}_G$ contains the edge (T_2, T_3). Finally, deleting (b, c) and inserting (d, e) changes T_3 into T_1, so they are adjacent, as well. Figure 3 shows the entire graph $\mathcal{T}_G$.