Fundamental Algorithms - Spring 2018
Homework 11

Daniel Rivera Ruiz
Department of Computer Science
New York University
drr342@nyu.edu

1. Since the graph is complete, all the vertices are in the adjacency list of the root $r = 1$ and therefore will be added to the priority queue Q during the initialization with $\pi(j) = 1$ and $k(j) = (j-1)^3$ for $2 \leq j \leq n$.
   At the first iteration of the algorithm, the minimal element in Q is 2 because $k(j)$ is a strictly increasing function[1]. After removing 2 from Q and adding it to S, we have to update π and k for all the values in Q because they are all in $adj(2)$ and they are all closer to 2 than they are to 1. Therefore, we have to make $\pi(j) = 2$ and $k(j) = (j-2)^3$ for $3 \leq j \leq n$.
   Following this intuition, at the i^{th} iteration of the algorithm the minimal element in Q will be i and after extracting it Q must be updated as follows: $\pi(j) = i$ and $k(j) = (j-i)^3$ for $i+1 \leq j \leq n$.
   (a) Under these conditions for a graph with n vertices where $n = 500$, the first 211 elements inserted in the MST have to be $\{1, 2, 3, \ldots, 211\}$.
   (b) After inserting the 211^{th} element and updating π and k for the remaining elements in Q we will have $\pi(309) = 211$ and $k(309) = (309-211)^3 = 98^3 = 941192$.

2. The following table shows the evolution of the `Extended-Euclid` algorithm to calculate d, x and y. Due to the recursion, the columns a, b, q and r are calculated from the top down, and afterwards d, x and y are calculated from the bottom up.

a	b	q	r	d	x	y
144	89	1	55	1	34	-55
89	55	1	34	1	-21	34
55	34	1	21	1	13	-21
34	21	1	13	1	-8	13
21	13	1	8	1	5	-8
13	8	1	5	1	-3	5
8	5	1	3	1	2	-3
5	3	1	2	1	-1	2
3	2	1	1	1	1	-1
2	1	2	0	1	0	1
1	0	-	-	1	1	0

   At the end of the algorithm we get $d = 1$, $x = 34$ and $y = -55$, which satisfies $d = ax + by \Rightarrow 1 = (144)(34) + (89)(-55)$.

3. To solve $\frac{311}{507}$ in $\mathbb{Z}_{1000}$ we observe that $gcd(507, 1000) = 1$ and therefore 507 has a multiplicative inverse in $\mathbb{Z}_{1000}$. The problem reduces to finding $y \equiv 507^{-1} (mod\, 1000)$ and calculating $311y \,(mod\, 1000)$.
   To find y, we use `Extended-Euclid` with $a = 1000$ and $b = 507$:

[1] In the definition of k we assume $j > i$ in order to assure $(j-i)^3 > 0$. This is a necessary condition since the definition of the MST problem requires that all weights w be greater or equal than 0.

a	b	q	r	d	x	y
1000	507	1	493	1	181	-357
507	493	1	14	1	-176	181
493	14	35	3	1	5	-176
14	3	4	2	1	-1	5
3	2	1	1	1	1	-1
2	1	2	0	1	0	1
1	0	-	-	1	1	0

So we get $1 = (1000)(181) + (507)(-357)$ and therefore $y = -357$ in $\mathbb{Z}_{1000}$. Finally we calculate $(311)(-357) \equiv -111027 \equiv -27\,(mod\,1000)$:

$$\frac{311}{507} = -27 \quad \text{in } \mathbb{Z}_{1000}$$

4. We observe that 101 and 103 are both prime numbers and therefore $gcd(101, 103) = 1$. Under this condition, we can use the *Chinese Reminder Theorem* to solve the system:
   (a) First we need to calculate $y \equiv (101^{-1})(59 - 34) \equiv (101^{-1})(25)\,(mod\,103)$.
   (b) To find 101^{-1} in $\mathbb{Z}_{103}$ we use `Extended-Euclid` with $a = 103$ and $b = 101$, which returns $d = 1$, $x = -50$ and $y = 51$, so $101^{-1} = 51$ in $\mathbb{Z}_{103}$.
   (c) We get the value of $y \equiv (51)(25) \equiv 1275 \equiv 39\,(mod\,103)$.
   (d) Finally, we replace the value of y in $x = 34 + 101y = 34 + (101)(39) = 3973$.

   The solution to the original system is given by

$$x \equiv 3973\,(mod\,101 \cdot 103) \Rightarrow x \equiv 3973\,(mod\,10403)$$

5. To compute 2^{1072} in $\mathbb{Z}_{1073}$ we follow the modular exponentiation algorithm:
   (a) Write $B = 1072$ in binary form: $1072 = 1024 + 32 + 16 = 2^{10} + 2^5 + 2^4$.
   (b) Repeatedly square $A = 2$ in $\mathbb{Z}_{1073}$:

i	2^i	2^{2^i}	2^{2^i} in $\mathbb{Z}_{1073}$
0	1	2	2
1	2	4	4
2	4	16	16
3	8	256	256
4	16	65536	83
5	32	6889	451
6	64	203401	-469
7	128	219961	-4
8	256	16	16
9	512	256	256
10	1024	65536	83

   (c) Finally we have:

$$2^{1072} = 2^{1024+32+16} = 2^{1024} \cdot 2^{32} \cdot 2^{16}$$
$$2^{1024} \cdot 2^{32} \cdot 2^{16} \equiv (83)(451)(83) \equiv (451)(451) \equiv -469\,(mod\,1073)$$
$$2^{1072} \equiv -469\,(mod\,1073)$$

   Since $2^{1072} \not\equiv 1\,(mod\,1073)$ we can conclude that 1073 is **not** a prime number. In fact, the prime factors of 1073 are 29 and 37.

6. When a node has itself as a parent, e.g. $\pi(v) = v$, it means that at that point in the algorithm the node v is the root of the component that contains it. Moreover, the size of the root is equal to the number of nodes in the component, i.e. $size(v) = 35$ means that the component where v is the root has 35 nodes (including v).

In the case where $\pi(v) = u \neq v$, $size(v) = 35$ means that at some point in the past, when v was its own parent and therefore a root, its component had 35 nodes. At this point, however, the actual number of nodes in v's component (where u is the root) is given by $size(u)$.
During the course of Kruskal's algorithm, $\pi(w)$ can take at most two values for each vertex w. The intuition behind this is that $\pi(w)$ will only be updated if two conditions are met: 1) if w is the root of its component (otherwise `sliding-down-the-banister` will just keep going past w) and 2) if $size(w) < size(v)$, where v is the other root being consider in the iteration. If at any point both conditions are met, $\pi(w) \leftarrow v$ will be updated. Once this happen (if it happens) it will never happen again, because w will no longer be a root and the first condition will never be satisfied.
During the course of Kruskal's algorithm, $size(w)$ can take at most V values, where V is the number of vertices in the graph. It is clear that $size(w)$ cannot be larger than V because it would mean that there are more than V nodes in w's component. To explore the extreme case, let us consider a graph G with edges of the form $e_i = \{1, i\}$ (for $2 \leq i \leq V$) that satisfies $w(e_i) < w(e_j)$ if $i < j$. In such a graph we will get an update of the form $size(1) \leftarrow 1 + size(1)$ for every edge, so $size(1)$ will take all the values from 1 to V.