Tim Cheeseman
CS 260-003
Summer 2014
Assignment 7 Sample Solution

5.20)

In a pseudo-python, using the MFSET operations from the book:

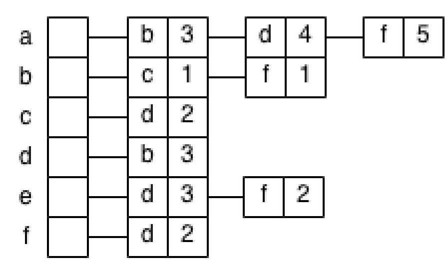
```
def compute_equivalent_states(n, transitions):
  for i in [1 ... n]:
    INITIAL(S_i, i)

  # compare each state to each other state
  for i in [1 ... n]:
    for j in [1 ... n]:
      if i == j:
        continue # no need to merge state with itself
      elif (i % 2) == (j % 2) # both accepting (even) or unaccepting (odd)
          and transitions[i, 0] == transitions[j, 0]
          and transitions[i, 1] == transitions[j, 1]
        # merge set containing i with set containing j
        MERGE( FIND(i), FIND(j) )
```

6.1)

a)

	a	b	c	d	e	f
a		3		4		5
b			1			1
c				2		
d		3				
e				3		2
f				2		

b)



We can trivially create a mapping function F(x) which maps a-f to array indices (e.g. F('a') = 0).

3)

Each edge in the undirected graph is incident to exactly two vertices. The degree of a vertex is the number of edges incident on that vertex. Therefore, when we sum the degrees of each vertex, we are counting each edge twice.

4)

Each edge in the directed graph leaves from exactly one vertex and arrives at exactly one vertex. The sum of the in-degrees of each node will therefore be equal to the number of edges, and the same is true for the out-degrees.