**Homework 3**

*Ilai Fallach, 200924751 and Orr Mandelbaum 200612190*

# Question 1

Pseudo code for K-Means in MapReduce paradigm:

**map\_fn**(k, centers, point):

new\_k = find\_nearest\_center(centers, point)

return new\_k, point

end

**reduce\_fn**(k, points):

new\_center = find\_new\_center(points)

cost = compute\_cost(new\_center, points)

return new\_center, points, center

end

**main**():

step\_delta\_threshold = 1

points = load\_points()

centers = initialize\_random\_centers(points)

step\_delta = step\_delta\_threshold + 1

last\_cost = None

while step\_delta > step\_delta\_threshold:

centers, points, cost = \

run\_map\_reduce\_job(centers, points, map\_fn, reduce\_fn)

if last\_cost is not None:

step\_delta = last\_cost – cost

last\_cost = cost

end

return centers, points

end

# Question 2

Pseudo code for CheckClique in MapReduce paradigm:

**map\_fn**(k, v):

yield 1, 1

end

**map\_fn\_2**(k, v):

d = v.split(“->“)

neighbours = d[1].split(“ “)

yield len(neighbours), 1

end

**reduce\_fn**(k, v):

return sum(v)

end

**main**():

graph\_text = load\_graph()

num\_vertices = run\_map\_reduce\_job(graph\_text, map\_fn, reduce\_fn)

result = run\_map\_reduce\_job(graph\_text, map\_fn\_2, reduce\_fn)

if num\_vertices – 1 is in result and \

result[num\_vertices – 1] == num\_vertices:

return True

else

return False

end

end

# Question 3

Pseudo code for Pseudo-Synonyms Detection in MapReduce paradigm:

**map\_fn**(k, v):

words = v.split(“ “)

yield (words[0], words[2]), words[1]

end

**reduce\_fn**(k, v):

pairs = compute\_pairwise\_combinations(v)

return pairs

end

**map\_fn\_2**(k, v):

first, second = v

if first > second:

second, first = first, second

yield (first, second), 1

end

**reduce\_fn\_2**(k, v):

return sum(v)

end

**main**():

text = load\_text()

pairs = run\_map\_reduce\_job(text, map\_fn, reduce\_fn)

pairs\_count = run\_map\_reduce\_job(pairs, map\_fn\_2, reduce\_fn\_2)

return pairs\_count

end