1. (20 points) Let $JS(S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}$ be the Jaccard similarity between two sets $S_1$ and $S_2$. Prove that $f(S_1, S_2) = 1 - JS(S_1, S_2)$ is a distance measure, that is, $f(\cdot)$ satisfies the following properties

(i) $f(S_1, S_2) = f(S_2, S_1) \geq 0$ (5 points)

(ii) $f(S_1, S_2) = 0$ if and only if $S_1 = S_2$ (5 points)

(iii) $f(S_1, S_3) \leq f(S_1, S_2) + f(S_2, S_3)$, $for\ any\ S_1, S_2, S_3$. (10 points)

2. (10 points) Build an FP-tree for the following transaction database. Sort items in support descending order. Draw the FP-tree.

| Transaction ID | Items |
|---|---|
| 1 | HotDogs, Buns, Ketchup |
| 2 | HotDogs, Buns |
| 3 | HotDogs, Coke, Chips |
| 4 | Chips, Coke |
| 5 | Chips, Ketchup |
| 6 | HotDogs, Coke, Chips |

3. (15 points) Consider computing an LSH using $k = 160$ hash functions. We want to find all object pairs which have Jaccard similarity at least $t = 0.85$. Suppose we use the $(r, b)$-way AND-OR construction, which means that a pair of documents with similarity $s$ is considered as a candidate pair with probability $1 - (1 - s^r)^b$. Choose the best $r$ and $b$. Justify why your choice is the best.

4. (55 points) Download the file "trans.txt", where every line is a transaction represented by a set of item ids.

(1) Implement the Apriori algorithm to find all frequent patterns under different settings of the minimum frequency (minimum support/#transactions). Vary the minimum frequency *minFreq* as 0.0001, 0.0002, 0.0003, 0.0004 and 0.0005. Report the number of frequent patterns, as well as the number of size-$k$ frequent patterns for each size $k$ with at least one frequent pattern, under each setting of *minFreq*. (35 points)

(2) Try to optimize your algorithm using acceleration techniques. Try to make your algorithm finish computing for the task in (1) within 10 mins. Explain each specific acceleration you adopt by providing a running time comparison between adopting the acceleration and not adopting the acceleration. (15 points)

(3) You also need to submit your code. You can use whatever programming languages you like. Please provide a readme file describing how to run your code. The TA will run your code to check your algorithm's running time. (5 points)