Assignment 4: RFID Scanning

CPSC 441 Fall 2021

Kathryn Lepine 30044629

Output Summary

Summary Table Starting from the Leaf Node

Basket	Leaf Collision	Leaf Idle	Leaf Success	Leaf Total Time Slots	Leaf Success Rate	Leaf Actual Time (microsec)
Customer1	0	1012	12	1024	1.17%	4
Customer2	0	974	50	1024	4.88%	4
Customer3	0	892	132	1024	12.89%	6
Customer4	0	874	150	1024	14.65%	8
Customer5	0	847	177	1024	17.29%	4
Customer6	0	695	329	1024	32.13%	9
Cusomter7	0	583	441	1024	43.07%	13
Cusomter8	0	567	457	1024	44.63%	11
Customer9	0	591	433	1024	42.49%	10
Customer10	0	653	371	1024	36.23%	12

Summary Table Starting from the Root Node

Basket	Root Collision	Root Idle	Root Success	Root Total Time Slots	Root Success Rate	Root Actual Time (microsec)
Customer1	17	6	12	35	34.29%	11
Customer2	70	21	50	141	35.46%	16
Customer3	164	33	132	329	40.12%	24
Customer4	191	42	150	383	39.16%	33
Customer5	225	49	177	451	39.25%	28
Customer6	378	50	329	757	43.46%	41
Cusomter7	484	44	441	969	45.51%	66
Cusomter8	498	42	457	997	45.84%	40
Customer9	478	46	433	957	45.25%	38
Customer10	421	51	371	843	44.01%	47

1. Which of the customers has the most items? How many items do they have?
    - Customer 8, they have 457 items.
2. When starting at the **leaf level** of the tree, which basket of goods takes the most time to scan? How many time slots does it require?
    - They all take the same amount of time to scan. The total amount of time slots is 1024 for each customer, which is the number of items available. For my actual time of program execution customer 7 took the longest with 13 microseconds.

3. When starting at the **root level** of the tree, which basket of goods takes the most time to scan? How many time slots does it require?
    - Customer 8 takes the longest time to scan, taking 997 time slots.
4. When starting at the root level of the tree, which basket of goods takes the **least time** to scan? How many time slots are needed?
    - Customer 1 takes the least time to scan, taking 35 time slots.
5. When starting at the root level of the tree, which basket of goods generates the **most collisions** during scanning? How many collisions occur?
    - Customer 8 causes the most collisions, with 498.
6. When starting at the root level of the tree, which basket of goods generates the **highest proportion of successful slots** (i.e., efficiency) during scanning?
    - Customer 8 has the highest proportion of successful slots, with 45.84%.

User manual

On line 66 of the code, you may change the int variable k. On line 76 of the code, you may change the name of the txt input file that is being read. Do make sure the input file is in the same folder before running the program. To compile the program simple type "g++ -o 4assignment 4assignment.cpp" on the command prompt and enter. To run the program type "/.4assignment" on the command prompt and enter. You will then see the output for both the leaf search and root search for the input txt file.

Summary

Multiple access protocols (MAC) are used to determine for a given channel, who can use the channel. If more than one sender is using the channel, then we will get a collision. If no one is using the channel we will get an idle. Lastly, if one uses the channel individually, they will get a success, if their message arrives complete to the receiver. In this assignment we use the adaptive tree walk protocol to allocate who can use the channel accordingly. Let us analyze some of the results from the assignment. The most general observation we can make is that the root search is much better if we do not have lots of items and the items are spread out to different branches. We can see the success rate from the root search are all over 34%, whereas the leaf search gets as low as 1% on a particular search. Out of the 10 customers analyzed the leaf search gave an average efficiency of 25% and the root search gave an average efficiency of 41%. We can say the root search takes less time than the leaf search. The highest time slot for root search was 997 for customer 8, whereas for leaf search every search simply took the number of items, 1024 each time. Let us touch on the point mentioned above about the root searches being higher efficiency if the items on the branches are far apart. If items are on branches that are far from each other then there will be less collisions and therefore less time slots. For example, if we have two items which are 1 and 1023, we will get 1 collision at the root and then 2 successes directly after. However, if we change the example to have items 1 and 2, then we will have collisions all the way down until the end of the tree, not to mention the idles that will be accumulated from checking the branches to the right. Thus, it is important that for the root search to be as efficient as possible there must be minimum items, with them on as far apart branches as possible. Although we do not have any customers with over half the items, we can extrapolate the data and see that the leaf search would see an improvement in time slots whereas the root search would see a decline in performance of

time slots. This is because if a basket has over half the items, then number of collisions would increase drastically. On the other hand, the leaf sort is a linear increase in efficiency based upon the number of items the customer has. Furthermore, we are thinking of this problem in terms on RFID scanning, in which case we would assume that not many customers would be buying a large number of items in the store. In fact, it would probably be safe to assume that the majority of customers would buy a very small percentage of items in the store. To emphasize this point further we take a look at customer 1 who only has 12 items, in the root search it takes just 35 times slots whereas in the leaf search it takes 1024 time slots. Therefore, in the case of RFID the root search would make a lot more sense because of the small batches of items being bought. From our 10 customers the average time slots from root search was 586 and the average time slots for leaf search was obviously 1024. It is hard to definitively say whether the difference between these two average time slots is large. However, what we can say is that when the number of items in the store increases the time slots needed for leaf search will also increase. This is important to note because many stores do not have just 1024 items, they could easily have 10,000 items in their store. So, in terms of scalability the root search is much more practical.