Logarithms and Search

Qiyam Tung

October 26, 2011

1 Search

How fast can you search for someone's phone number from a phonebook? We will use this activity to find out how the time it takes to search for a phone number is related to logarithms.

For this activity, you should get into groups (because we don't have enough phonebooks) and find the phone number of the first entry for each a category. For example, if I asked you to search for the first number under *Prosthetic Devices*, you will find that under that category, the first entry is *APOS Arizona Prosthetic Orthotic Services* and the phone number to be 229-0622.

With that in mind, let's have a contest! Which group can find the phone numbers the fastest? Find the first entry's phone number for each of the following categories:

1. Hotels & Other Accomodations (Adobe Fountain Suites Hotel 887-6959)
2. Cable Televsion (A ACE DISH 877-616-5651)
3. Mats and Matting (ALSCO 623-0581)
4. Packaging Materials (Associated Bag Company 800 442-9129)
5. Self Defense (The Ultimaa Martial Arts Center 744-4591)

2 Algorithm

Obivously, you all tried to find a name as fast as possible. **Can you describe the method that you used to find the name? Explain why you chose that reason.** You can reason about it by asking yourself why you didn't use other method. For exmaple:

- Did you search by starting at page 1? Why not?
- Did you search by flipping to a random page? Why not?

2.1 Your Search

Most likely, what you did was search by estimating where the name would be in the book. If you flipped to the wrong page, you'd again guesstimate where the name is.

However, if I asked you to write down your method and prove to me that it is both **fast** and **guaranteed to find the name** (or tell us if it isn't in the phonebook), it would be difficult to prove the second condition. This is mostly because you "guesstimated" the location of the name. Instead, let's look at a very similar method that is easier to analyze.

2.2 The search method: binary search

Binary search can be thought of as dividing your search space into smaller and smaller sections. Let's assume the book has 100 pages. Here's how you would do binary search. For simplicity's sake, let's assume there is only one name per page.

1. Given a phonebook and name to search.
2. Look in the middle of the remaining pages to search.
3. Check the name on the page with the name you're looking for.
4. If it's before the name you're looking for, consider only at the right half. Go to step 2.

5. Otherwise, if it's after the name you're looking for, consider only the left half.

6. Otherwise, if it's the name you're looking for (or if there aren't any more names to search), you're done.

3 Analysis and Logarithms

3.1 Visualization

You can visualize your search method to look like this the figure below. Assuming there are 15 pages, Linda is on page 7, Fred is on page 4, Bonnie is on page 2, and Aaron is on page 1. So, if you "push" all the boxes down onto one line, they would be alphabetically sorted from left to right.

This helps understand why binary search is fast. For example, say that we're looking for Aaron, then our search would start in the middle of the book. We'd first find Linda and realize that Aaron is before Linda. So, we would move down to the left of Linda and find Fred. We then look to the left of Fred and find Bonnie and then move down to the left once more to find Aaron. In total, we had to look at 4 names before we found Aaron. If we were looking for Bonnie, we would have only need to search for 3 names before we found her.

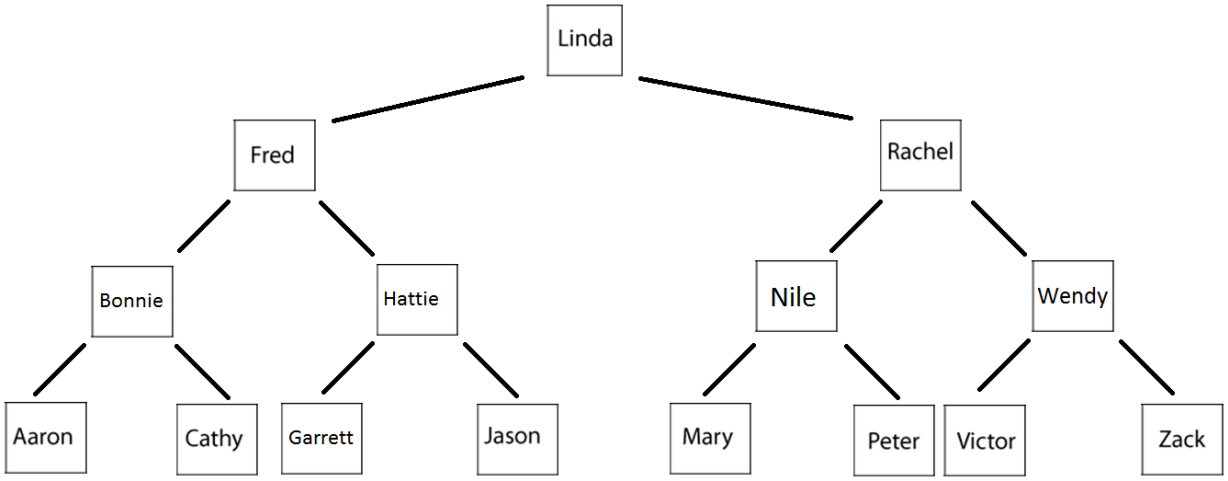

Figure 1: Visualizing your search method on a phonebook.

With this picture in mind, answer the following questions

1. How many names would you have to check to find Zack?

2. How many names would you have to check to find Wendy?

3. How many names would you have to check to find Nile?

4. How many names would you have to check to find Fred?

5. Which name(s) would require the least amount of checks?

6. Which name(s) would require the most amount of checks?