Homework #3 (LCP searches)

Problem #1. Given (e.g.,) 1024 LCPs.

a) We have shown a bit-splitting technique that works on our LCP number's representation in binary notation. There is nothing special about base 2; we could have equally well built our bit-splitting sets based on base 3 or base 4. Assume that there is exactly one critical path, that we use binary subdivision to isolate the driver once we have a hit in a bit-splitting set, and that we only do the subdivision on one splitting set. For 4096 LCPs, compute the actual number of runs needed for base 2, base 3 and base 4 (remember that you cannot have a fractional number of sets or a fractional number of test runs). Which base is thus best?

b) We have assumed that subdivision uses a binary tree. In fact, if there is exactly one hit to find, then binary is optimal for subdivision. Is binary subdivision still optimal if there are many hits? Argue your case either way. For the extreme case where every single LCP in your subdivision tree is a hit, can you come up with a faster strategy than binary subdivision?

Problem #2. Assume that there are 16 LCPs, and that we have one critical path from driver #4 to receiver #10. Which splitting sets will find this path (e.g., bit #1=0)? Pick one of the splitting sets and show the path through subdivision to find the driver – which children will get hits? Considering that the driver will be found from more than one splitting set, how many total test runs will be needed to find the driver?

Finding the same driver multiple times (and, if fact, then finding its receiver(s) multiple times) is clearly inefficient. If you do the trick we talked about in class to avoid this, then how many test runs will be needed?

Problem #3. Assume we have to do LCP searches on 100 chips. After 50 chips, we find that paths from LCP #17→23, #50 →40, and #217 →12 are found on nearly every chip (though a few other paths are also found on some chips). Can you propose a new splitting set to add (and assume that we test this new splitting set first), so that the remaining 50 chips get tested as fast as possible? What if the path #49→17 were also found frequently? Assume again that we use the trick we talked about in class so as to avoid finding the same driver multiple times.

Please turn in your assignment via the Provide cgi interface. You can use any reasonable format (including writing your answers by hand and taking a picture of the page with your phone).