COP 5536 / AD 711R Advanced Data Structures

Instructor: Dr. Sartaj Sahni
Spring, 1999

Exam 3
Closed Book
1 Hour

April 1999
Final

Name: _____

SSN: _____

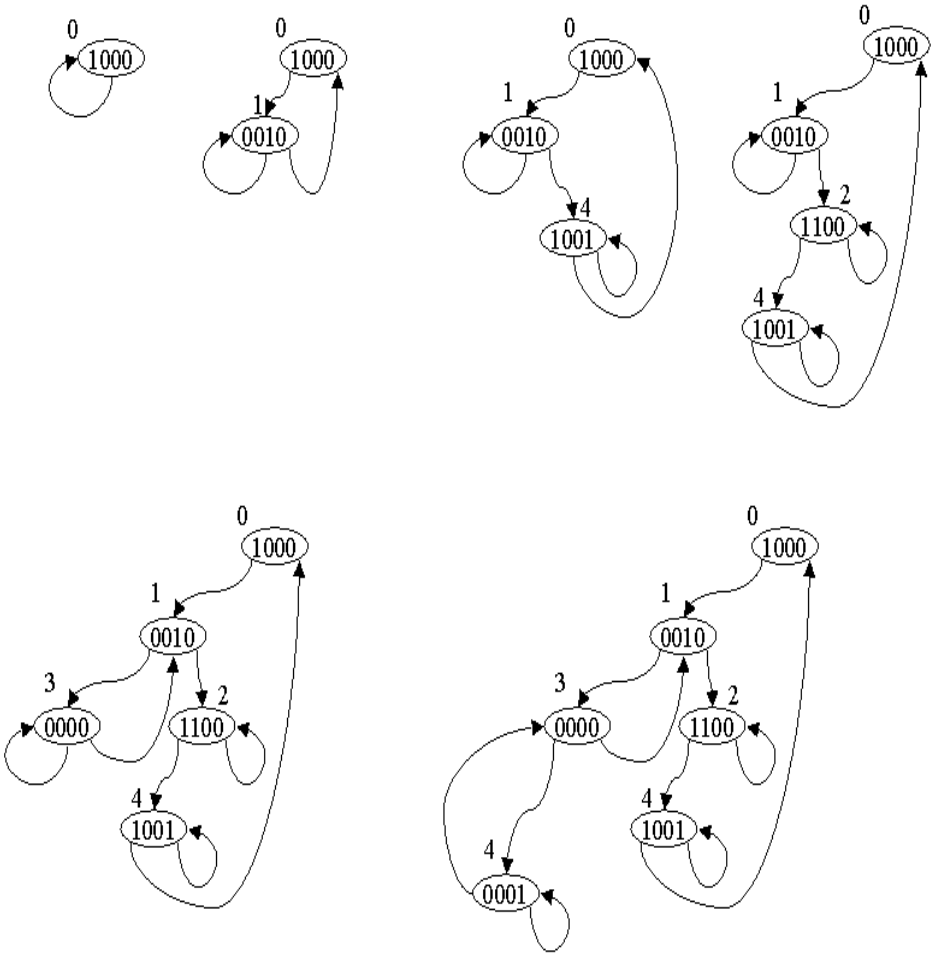
Site Number: _____

NOTE: All answers will be graded on correctness, efficiency, clarity, elegance and other normal criteria that determine quality. The points assigned to each question are provided in parentheses.

1. (10) Insert the following keys into an initially empty Patricia.

`1000, 0010, 1001, 1100, 0000, 0001`

Draw the Patricia following each insertion.



2. (10) B-tree question

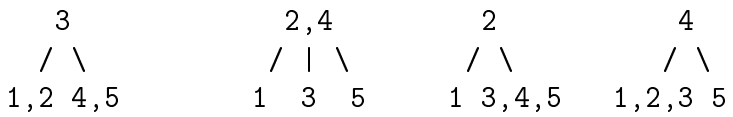
(a) (5) Assuming that a B-tree (of height h) is kept on disk and one node may be fetched at a time, how many disk accesses are needed to insert a new element? Give an explanation of how you arrived at that number.

Searching for the insertion position takes h accesses (when the key is not there). We need h splits as we go back up the tree (cascading split in worst case). Each split takes 2 disk accesses (to write the new nodes). When we get back to the root and split it, there are 3 accesses: 2 for splitting the node (already accounted for) and 1 for writing the new root.
Therefore, the total number of disk accesses is h + 2h + 1 = 3h+1.

(b) (5)Show all B-trees of order 4 or less that contain the keys 1, 2, 3, 4, 5.

There are not any B-trees of order 2 here.
B-trees of order 3:

B-trees of order 4:



There are 6 trees. Though some of the trees look the same they are different, because they are of different orders.

3. (10) Give pseudo code to insert a string into a Trie. Use the sampling function, sample(x,i) = ith character of x.

```
insert(key){
  key = concatenate(key, ' ');
  **p = &root;
  int i = 1;
  for(; p is a branch node; p = &((*p)->link[c]),i++){
    c = ith character of key;
  }
  if(*p == NULL){ *p = new node(key); }
  else{
    if(p->key == key){ return; }
    else{
      *temp = *p;
      *p = NULL;
      for(; ; p = &((*p)->link[c]),i++){
        *p = new branch_node();
        c = ith character of key;
        c2 = ith character of temp->key;
        if(c != c2){
          (*p)->link[c] = new node(key);
          (*p)->link[c2] = new node(temp->key);
          return;
} } } } }
```

4. (9) Give an application (and describe how the structure is used to satisfy the application needs in 2-3 sentences each) for:

   (a) Suffix Tree:

   The problem of determining if string P is a substring of S is handled by: build a suffix tree from S and then search for P in the suffix tree (upto depth |P|).

   (b) Bloom Filter:

   A Datbase master file is too large to fit in memory, so a differential file is often used. A bloom filter avoids search time between the files by attempting to answer the question: "is the key present in the differential file". The bloom filter will give 1 of 2 answers to the posed question: No and maybe.

   (c) Quad Tree: For image processing we need to look at points in space. A quad tree helps build relevant boundaries around points (cells) allowing for fast processing.

5. (10) Describe how to use a Priority Search Tree to implement a memory management scheme which includes methods to allocate memory by best-fit and first-fit (a special case of resource allocation). You need to give any relevant functions and their purpose.

The following is used for parts a and b:

let m = memory in the system
let x = memory size
let y = address of memory
since we cannot repeat values we use a transformation function as discussed in class,
x = (size of memory request)*m + memory address,

(a) Describe a best-fit scheme (smallest available block of memory whose size is $\geq$ the request):
min x in rectangle (L,R,top) min to the left

(b) Describe a first-fit scheme (smallest memory address whose size $\geq$ the request):
min y in rectangle range(x0,x1) bottom most point