308-420A Secondary storage Algorithms and Data Structures

Supplementary Questions and Exercises

Logarithmic Files

Section 1.2–4, Logarithmic Files

1. A B-tree of height 6 contains 170,000 nodes with an average of 18.8 records per node. How many splits occurred when the B-tree was created?

   (a) 169,999

   (b) 6

   (c) between 1 and 6

   (d) 0.059 (1/18.8)

   (e) None of the above.

2. An alphabetical (as opposed to binary) kd-trie is built in three dimensions in such a way that each dimension in turn is used, cyclically, for any search. A search is being made for the three keys, `RELATIONAL, INFORMATION` and `SYSTEMS`. What is the tenth letter examined, and what is the discriminator at this point?

   (a) A, first key.

   (b) L, first key.

   (c) O, second key.

   (d) S, third key.

   (e) None of the above.

3. Arrange the points `(37,18), (23,23), (32,42), (12,8)` in two-dimensional Z-order.

   (a) `(37,18), (23,23), (32,42), (12,8)`

   (b) `(12,8), (32,42), (23,23), (37,18)`

   (c) `(12,8), (23,23), (37,18), (32,42)`

   (d) `(12,8), (23,23), (32,42), (37,18)`

   (e) None of the above.

4. What philosophy should guide the design of dynamic file stuctures, and how do B-trees implement that philosophy? Name two other dynamic file structures discussed in the course that implement the same philosophy in different ways.

5. What two general directions may be taken to improve the performance of the original B-tree algorithm. Give some details.

6. The worst-case cost for retrieving a single record from a particular kind of tree is

$$1 + \log_{\lceil f/2 \rceil}(\frac{N+1}{2}).$$

   What does this tell about the tree? Explain the symbols f and N.

7. What is *Z-ordering* and how could it be combined with hashing?

8. The following four pairs of values can be Z-ordered in two different ways in two dimensions. What are they? Give the actual sequences.

   (9, 7) (5, 8) (11, 1) (2, 15)

9. In C, write the data structure required to represent the node of a homogenous B-tree of order 20 with integer pointers and 80-byte character strings of data (for which the breakdown into fields is not specified).

10. Which of the three rules for the B-tree of order f ensures balance? How is this rule implemented?

11. To improve B-tree performance, what essential quantity must be increased? Discuss two different approaches to increasing it.

12. Given a B-tree implementation, how would you use it in an application requiring high *symmetry*?

13. Show the binary trie that represents *all* suffixes of the bit string

    `11100101`

    (A *suffix* of a string s is a substring of s ending at the end of s.) Discuss any problems you encounter.

14. A binary kd-trie in two dimensions is used to store the following points.

    `(0,0) (1,1) (2,2) (3,3) (4,4) (5,5) (6,6) (7,7)`

    Write down the order in which these points would appear in a preorder traversal of the trie.

15. A homogeneous B-tree of order 3 is built from the records `A, B, C, D, E, F, G`. Show the best-case B-tree and the worst-case B-tree that can result.

16. A homogenous B*-tree of order 3 is built from the records `A, B, C, D, E, F, G` inserted *in that order*. Show the resulting B*-tree. (Consider that nodes overflow to the left.)

17. An inhomogenous B-tree of order 3 (at all levels) is built from the records `A, B, C, D, E, F, G` inserted *in that order*. Show the resulting B-tree.

18. Describe the performance of B-trees (in general) under requirements for "activity" and "volatility".

19. The following records have two fields each, and are inserted, in the order shown, into a binary kd-tree. The discriminator starts at the first field. Matches descend to the *left*. Show the resulting tree.

    (2,2), (1,2), (3,2), (2,1), (2,3)

20. The following records have two fields each, and are inserted, in the order shown, into a binary kd-*trie*. The discriminator starts at the first field. Matches descend to the *left*. Show the resulting trie.

    (2,2), (1,2), (3,2), (2,1), (2,3)

21. Show four different binary trees on the keys `A, B, C, D` which are completely degenerate, that is, which require O(N) search time.

22. Show all possible B-trees of order 3 that can be built from the keys `A, B, C, D`.

23. If we change from a homogenous B-tree to an inhomogenous one for the same data, how is the order of the B-tree affected, and why is this an improvement? What is the corresponding disadvantage, and is it serious?

24. *Explain* two techniques whereby B-trees can be useful for multidimensional data.

25. (a) How many nodes does a completely full B-tree of order f have at level $i (0 \leq i < h)$? Show why.

    (b) How many records does a completely full B-tree of order f and height h hold in total? Show why.

26. The following is the Z-order value of a data point.

    `011001011011`

    a) If it is a 2-dimensional point, what are its coordinates? b) If 3-dimensional? c) If 4-dimensional?

27. Homogenous B-trees of order 3 are built from the following sequences of data. What is the load factor of the resulting trees? (Neglect pointer storage.) a) `1, 3, 4, 6, 8, 2, 5, 7` b) `1, 2, 3, 6, 7, 4, 5`

28. Using the formulas for homogenous B-trees of order 3, determine a) the smallest possible height for 8 records and b) the biggest possible height for 7 records.

29. What are two ways to make B-trees more efficient? Briefly discuss the form taken by the improvement in each case.

30. What are two important differences between B-trees and kd-trees?

31. The points (3,7) and (4,0) are nearest-neighbours in Z-order, but $\sqrt{50}$ apart in 2-dimensional space. Here is another 1-dimensional ordering of 2-dimensional space, in which several pairs of nearest neighbours are at least $\sqrt{50}$ apart. a) What are all these pairs? b) Use this to compare briefly the two orderings as ways of representing symmetrical data. Is the second ordering symmetrical?

    The new ordering is given by $y+r(r+1)/2-q(q+1)/2$ where $r = x+y$, $q = \max(0, r-7)$, and x and y are the possible values of the two fields, $x, y \in \{$`0, 1, 2, 3, 4, 5, 6, 7`$\}$

    Hint. Draw a picture and neglect q.

32. A variant of Z-order is based on a kd-trie in which the discriminant cycles through x, then x again, then y ($xxyxxy\ldots$), where x and y are the two fields. Write down the order of the following sixteen points, $(x, y) =$ `(0,0), (0,1), (0,2), (0,3), (1,0), (1,1), (1,2), (1,3), (2,0), (2,1), (2,2), (2,3), (3,0), (3,1), (3,2), (3,3)` under this ordering, or draw the picture.

33. A Z-ordered file containing every possible record on fields $x, y \in$ `0, 1, 2, 3, 4, 5, 6, 7` is queried on the range $(x, y) =$ `(3,2)..(5,4)`. a) If only one probe ("direct" access), followed by nothing but scans (sequential accesses), is made, how many records will be retrieved? b) If only the 16 records are to be retrieved, how many probes (direct accesses) must be made? (As opposed to scans (sequential accesses).) Suppose the blocksize is 2.

34. What does the B-tree achieve that a binary tree cannot, and how?

35. How do we build a B-tree to guarantee a load factor of at least 66%? What new operation must be used when constructing such B-trees?

36. The key, "`Nat`", is stored in an inhomogenous B-tree of height 5. Searching for this key, we get a match the level below the root node. What must we then do to find the data associated with "`Nat`"?

37. A *full trie* is a trie which stores *all* key information in the paths. An *ordinary trie* has none of these paths from the leaves up to the first bifurcation above the leaf. A *Patricia trie* has *no* nodes of fanout 1.

    Which trie would be the best to use for indexing arbitrary substrings in large texts? Why?

38. Which of the above three tries would be best for supporting variable-resolution queries on spatial data? Why?

39. A kd-trie, represented as a full trie, holds all the data points, (x, y) for $x \in \{0, ..,7\}$ and $y \in \{0, ..,7\}$, with x being the first discriminator. Which leaf of the trie terminates the path describing (3,2)? Give its number, counting from (0,0) as number 1.

40. A full trie is stored on secondary storage in blocks with a maximum capacity of $\phi - 1$ nodes each. The trie is h levels high, including the root. Suppose the blocks all hold the maximum number of nodes.

    (a) Derive the expression for the number of *nodes* in the entire trie.

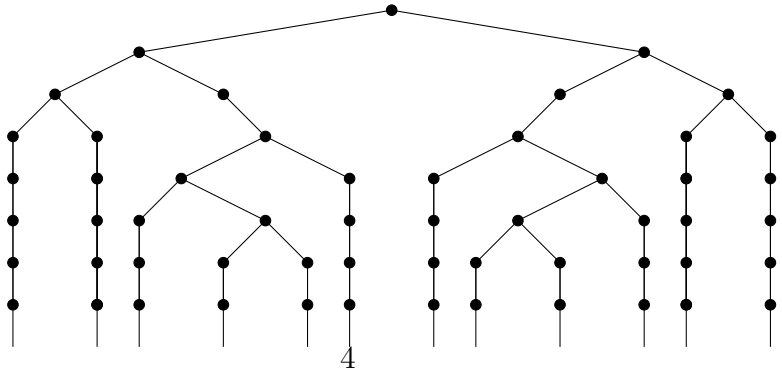
    (b) Write the sum that gives the number of accesses needed to read every *block* in the file, each read starting from the root block, and say in wor how you would simplify it.

41. Which of the three properties defining B-trees guarantees balance? State it in full.

42. Show the pointerless representation of the binary trie holding the numbers

    `0, 1, 2, 3, 4, 5, 6, 7`

43. The following trie is paged with four trie levels per page level. How many pages will it occupy, assuming the smallest reasonable pagesize?



44. What are the values for all the "top" and "bottom" counters according to the way you paged the trie in the previous question?

45. Considering the trie shown in question 42 as a 2-d trie, what are the coordinates of the points in 2-space? Suppose the vertical edges in the left half of the trie mean "`0`"; and those in the right half mean "`1`".

46. There are six different orderings of the data

    `1, 2, 3.`

    a) Which of these, when input to the insertion algorithm for a binary tree, will build a perfect tree? b) Which of the orderings will build a degenerate tree?

47. Show the inhomogeneous B-tree (B^+-tree) generated for the "records" (1 letter each) `T,H,E,Q,U,I,C,K,R,E,D,F,O,X`, entered in this order. Use a leaf blocksize of two records and a maximum fanout of seven for the internal nodes. Assume that the search algorithm looks to the left of a match in an internal node, and that duplicate entries are not re-inserted.

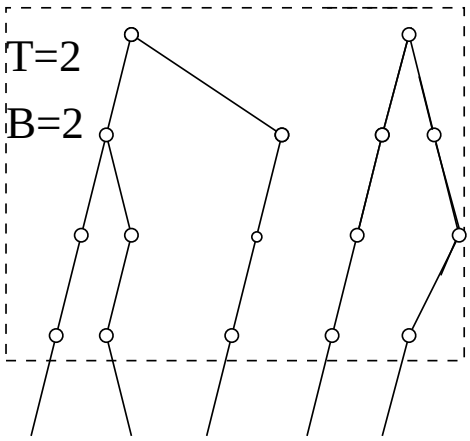
48. What is the advantage of an inhomogenous B-tree over a homogenous B-tree and why? Use the example of the previous question to support your argument.

49. Draw the trie corresponding to the following pointerless representation.

    ```
    11
    0110
    1001
    1111
    ```

50. Consider a trie which is a complete binary tree (i.e., height h and 2^h leaves). Beyond what height is the trie representation smaller (more compressed) than the source data stored in it? Why?

51. The path for the bitstring `1000` has been found in the page shown of a paged trie.



    The top and bottom counters for the next level are

page	a	b	c	d	e
T	0	2	4	6	9
B	0	4	7	10	14

Which of the five pages in the next level must be searched next?

52. The following data pairs are loaded into a two-dimensional kd-*tree* (a 2d-tree) in the order given. What is the height of the tree?

$$(0,0),(1,0),(1,1),(1,3),(3,0),(3,1)$$

Assume that matching values are sought to the right.

53. Sort the pairs in the previous question into Z-order.

54. In a 4×4 Z-order, (0..3)×(0..3), how many probes (direct accesses) are needed for the range query (1..2)×(1..2) if no extra points are to be scanned (sequential access)?

55. In constructing file structures for dynamic data, we have been able always to follow a single strategy. What is it? Illustrate this in two cases

56. Given the sequence of records (each abbreviated to one letter)

`T, Q, R, F, J, O, W, L, D`

show the homogenous B-tree of order 3 that results from inserting them one after the other, in the above order.

57. What is the load factor for the B-tree you built in the last question? (Neglect pointer storage in answering.)

58. What are two ways to increase the effective fanout of a B-tree without changing the size of the nodes?

59. The following hexadecimal sequence is a very short "text".

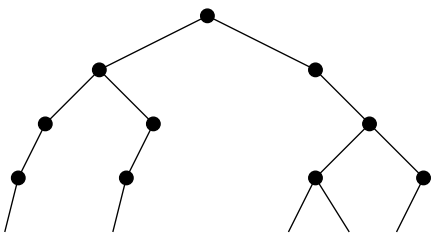
`6d6f636861`

Build a truncated binary trie for it with a branch starting at every eighth bit.

60. How many leaves are there in the trie whose pointerless representation is the following bit string?

`10010110111011111011`

61. What compression is achieved by the trie shown?



62. If the trie shown in the previous question were paged with two node levels per page level, what are the top and bottom counters for each page? (Page size is the smallest permitted power of 2.)

63. If the trie in the previous question were a two-dimensional kd-trie, a) what are the coordinates of the points it represents? b) what are the 1-dimensional values of these points in z-order (with a starting value of 0)?

64. An inhomogeneous B-tree of order 7 and load factor=100% has 49 data blocks (leaf nodes). How many accesses are required, on average, to find a record?

65. In the inhomogeneous B-tree of the previous question, the search keys of the records are the integers `0..146`. How many accesses are done by a range search which seeks records `27..48` inclusively?

66. A homogeneous B-tree of order 11 and height 4, has been loaded by records presented in such a way that it is as empty as possible. How many splits have been made?

67. How much compression is gained by the pointerless trie representation of any seven items of three-bit data?

68. In a paged trie, the root page contains the trie for data `0000,0001,0101,0111,1010,1011` and the four pages one level below the root have top pointers with values `0,1,3,5`, respectively. What will these top pointers become if the value `001101..` is inserted?

69. Given that `a` is hex `61` in ASCII, and the subsequent letters of the alphabet have subsequent values (`b` is hex `62`, etc.), show the *correct* PATRICIA trie for the text `mocha`: there should be one path to each of the five characters.

70. The points `(2,2),(2,4),(4,2),(4,5),(6,5),` and `(7,3)` are stored as a kd-trie, with the first of each pair as the first discriminator. In what order will they appear in a traversal of the kd-trie?

71. Draw a homogenous B-tree of order three using as keys the letters of `GEORGE III` (including the blank) and as data the sequence number of the letter (*e.g.*, `G 1, E 2, O 3, R 4, G 5,..`). Suppose that a duplicate occurrence of a key is considered to be larger than the earlier occurrence.

72. In a B-tree with duplicate keys (such as in question 71) of n nodes and of height h, how many accesses must be made to retrieve all duplicates of a given letter in the worst case?

73. In a homogenous B-tree of order four, the record `R` is to be added to a node containing `E G O`. Which record is moved up to the parent (or forms a new root)?

74. Data from comparison among large sequences is encoded in "trit"s, each with three possible values from the alphabet "<", "=" and ">". The data is represented either as simple strings in this alphabet or as a ternary trie, with fanout 3.

    What is the maximum compression that could be achieved by such a trie, of height h, over the simple representation of data? Is this better or worse than the maximum compression offered by a binary trie? (Hint: for the pointerless representation of the trie, note that 2 trits can hold everything 3 bits can.)

75. If all the nodes in the root page, R, of a paged trie need 2 Kbytes of storage, if all the 4-byte top pointers for the immediate descendent pages of R were stored in R, *and* if R and all of these immediate descendent pages each contain complete subtrees of nodes, how many bytes would we need for the root page?

76. Given that `0x3`d (not `0x3D`) is the hexadecimal for the ASCII byte representation of the digits d = `0`, .., `9`, draw the Patricia trie for all the sistrings in `02468`.

77. Sort the following data into the order given by traversing the 2d-trie representing it. Assume the discriminant cycle starts with the first field.

`(42,78), (66,93), (75,14), (11,99), (23,13)`

78. Draw a 4 by 4 Z-order curve and show on it a 2 by 2 range query which requires 4 probes.

79. What does the Montreal telephone book (hardcopy) tell us about the practical difference between linear and logarithmic search? Give algorithms in your answer.

80. What does the binary tree (RAM version) tell us about the practical difference between linear and logarithmic search? Give a construction algorithm in your answer.

81. Build a (ordinary, homogeneous) B-tree of order 4 from keys presented in ascending order. By the time the eighth key (`H` if you use `A, B, C, ..`) has been inserted, a) what are the smallest and largest load factors you encountered in the process (not counting space for pointers), and b) how many nodes have been created and what is the block number of the root node?

82. a) For a homogeneous B-tree of order f, how many records are there in total on level $l > 0$ for the cases with the highest and lowest possible load factors? b) For level $l = 0$?

83. How many bitstrings must we store in a trie of height h for the trie to need the same amount of storage as the original data? (Treat the trie as a ϕ-ary tree, and write an equation to be solved for x. Final solution not required.)

84. a) How many levels of trie nodes can fit into an 8Kb (kilobyte) block? b) If a file has 100 million 30-byte records stored in this trie, how many accesses are needed to fetch one of these records?

85. Rank the following three-dimensional points in ascending Z-order.

`(6,7,1), (7,5,8), (8,0,0)`

86. Suppose the *imbalance* of a tree is defined as the difference in path lengths:
(longest path from root to a leaf) − (shortest path from root to a leaf).
a) What are the *imbalance*s of a best-case full B-tree, a worst-case empty B-tree, a complete binary tree, and a degenerate binary tree? b) What is the biggest *imbalance* a binary tree of N entries can have?

87. Given that `a` is `61` in hex and `z` is `7a` in hex, draw the binary trie that represents all the letters from `a` to `z`.

88. a) How many bits are needed to store all the letters from `a` to `z`? b) How many bits are needed to store them as a trie in pointerless representation? (Use *your* solution to the previous question for this part.)

89. Is the datum `0110` present in the trie represented in pointerless form by

`11010101011111100110`?

If so, which leaf node matches it (give a number counting from left to right starting at 1). If not, give the number of the leaf immediately before where it would be if it were there.

90. a) How many 1-byte "pages" would be needed to store the trie for the following data (assume any counters needed are stored off-page)? b) How many page-levels are there, and how many node-levels per page?

```
0000  0001  0100  0101  1000
1001  1010  1011  1100  1111
```

91. If `0110` were added to the trie of the previous question, how many pages (still 1 byte each) would be needed?

92. Suppose the data of question 12 were represented by a kd-trie of two-dimensional points. Write the corresponding ten pairs of coordinates *in Z-order*.

93. If the kd-trie of the previous question were stored two records per page (a record consists of the two coordinates as key and whatever other data we might imagine to be associated with this key), how many pages will be accessed by the orthogonal range query ([1,2],[1,2]), *i.e.*, $1 \leq x \leq 2$ and $1 \leq y \leq 2$ where x is the first cooordinate and y is the second? Which page(s), counting the first page as 0?

94. A binary tree has 31 items in it, and each one is to be found, starting at the top each time. It is not specified how well balanced the tree is. a) How many comparisons must be done to find all items in the best case? b) How many comparisons must be done to find all items in the worst case? (You may show clearly how you would calculate the result without working out the numbers.)

95. A homogeneous B-tree of order 5 has 124 records in it, and each one is to be found starting at the top each time. a) How many accesses must be made to find all items in the best case? b) How many accesses must be made to find all items in the worst case?

96. a) Why is using an inhomogeneous B-tree worth giving up the possibility that a search may find the record before going all the way to the leaves, as in a homogeneous B-tree? In your answer, discuss expected depth of search in terms of h, the height of the tree, and in terms of fanout. b) What is the advantage of the inhomogeneous B-tree?

97. The lower-case letters of the alphabet have an ASCII representation starting with `61` (hexadecimal for `a`) and ending with `7a` (`z`). a) How many bits are required by the trie storing all 26 letters? b) What fraction is this of the bits needed for the letters, before they were organized into the trie?

98. A search in a paged trie yields a path which, at the bottom level of pages, is the third path in its page. This page has a bottom counter $B = 114$. Only the keys are stored on the trie, and the rest of the data, associated with these keys, is stored in a file of fixed-length records and blocksize $b = 40$. Which block of this data file should now be retrieved to get the rest of the data associated with the path just found? (Block addresses start at 0.)

99. A kd-trie stores the records
    `(0,0),(0,2),(0,3),(1,2),(1,3),(4,0),(4,3),(6,2),(6,4),(7,3),(7,7).`
    If this trie is traversed from all-`0`s path to all-`1`s path, in what order will it get these records?

100. If a kd-tree is loaded with the records of the previous question, in the order shown, how many comparisons must be made to decide where to put the last record? (Branch left, not right, on matches, unless the whole record matches.)

Section 1.3–4 (supplement), Logarithmic Files

1. A telephone book of 3.2 million 100-byte records is organized as a homogeneous B-tree of order 20 on a disk with a transfer time of 2 microseconds (μs.) per byte and an average seek time of 20 milliseconds (ms.). If the B-tree is organized to search by name, what are the best and worst times needed to find one entry? (Neglect storage for pointers.)

   (a) 23.8 to 119.0 milliseconds (ms.).

   (b) 23.8 to 166.6 milliseconds (ms.).

   (c) 142.8 to 166.6 milliseconds (ms.).

   (d) 144.0 to 190.4 milliseconds (ms.).

   (e) None of the above.

2. For a homogenous B-tree of order f and height h, what is the *greatest* number of records, N, that it can store? Show how to derive your answer.

3. Give a strategy for improving the performance of the B-tree described in the last question, and say (in words only) how it will affect the answer to that question.

4. Show that the expected search depth in a complete binary tree of height h is $h - 1$.

5. A homogenous B-tree of order 20 has 1,999,999 records. What is the largest number of accesses expected in a search for a single record.

6. Show that the expected depth of search in a complete binary tree is very close to $h-1$, where h is the height of the tree.

7. Derive closed expressions without Σ in them for a) $\sum_{k=0}^{h-1} \phi^k$ and b) $\sum_{k=0}^{h-1}(k+1)\phi^k$ (Tedious algebra is not necessary as long as you say what should be done.)

8. Consider the emptiest possible homogeneous B-tree of order 5 and height 2. Neglecting space for pointers, what is its load factor?

9. Consider two B-trees of height h, one ("$\mathcal{H}$") homogeneous, the other ("$\mathcal{I}$") inhomogeneous, with the same number of bytes per block. What is the major advantage of $\mathcal{I}$ over $\mathcal{H}$?

10. What are the best and worst costs for $\mathcal{H}$ and $\mathcal{I}$ of the previous question, for successfully retrieving a single record in the file? Also give the expected cost, in practice, for both.

11. r records are retrieved from a B-tree of height h and effective fanout ϕ. How many accesses are expected? Assume that the usage distribution is uniform, that $|\text{RAM}| = \mathcal{O}(h)$, and that the best possible algorithm is used.

    Work out the exact formula for $r = 2$

12. What is the data compression ratio for a complete binary trie of height h, over the original data?

The author gratefully acknowledges support from the taxpayers of Québec and of Canada who have paid his salary and research grants while this work was developed at McGill University, and from his students (who built the implementations and investigated the data structures and algorithms) and their funding agencies.