p4q3
MR

2 Data structures and algorithms 2003

(1) A application requires a hash table to hold up to 10^6 key-value pairs where both the keys and the values are integers. Carefully describe the two possible implementations: (a) Open Hashing using linked lists of key-value pairs outside the table and (b) Closed Hashing in which all key-value pairs are held within the hash table. [8 marks]

(2) Discuss how you would decide which method is most suitable for the given application. [6 marks]

(3) Assuming the table has exactly 10^6 entries and that keys, values and list pointers are all of size 4 bytes, and that you have allocated a total of 16 million bytes for the table (and hash chains), estimate the expected number of key comparisons necessary to locate an existing entry for (a) the open hash table and (b) the closed hash table. [6 marks]

```
ANSWER NOTES:

Hash tables are a standard part of the course.

(1a)
Space for hash chains = 3 words x 10**6
Allow between  10 x 10**5 words in the hash table.
Total space = 12 Mb + 4 Mb = 16 Mb

Average hash chain length = 1

Algorithm: (pseudo code)
          hashval = hash(key);
          p = &T[hashval];
          while(p) {
            if(p[1]==key) return p[2];
            p=p[0];
         }
         return NOTFOUND;

hash(key) could = abs(key) mod Tablesize

(1b)
Allocate 2 x 2 words x 10**6 for table
Requires 16 Mbytes.
The word pairs contain key-value pairs.
When 10**6 entries are in the table, the table is 1/2 full.
Expect 1/(1-1/2) (ie 2) probes to insert a new entry and about 1.5 probes
to lookup an existing entry.

Choose an unused key (eg 0) to mark empty entries.
Init the key fields in the table.

Alg:

hashval = 2 * hash(key)
while(T[hashval]) {
  if(T[hashval]==key) return T[hashval+1];
  hashval = (hashval + 2) mod Tablesize;  // see note
}
return NOTFOUND;
```

Note the secondary hashing function could be better.
eg something like:

```
hashval = (hashval + c) mod Tablesize
 where c = hash2(key) and
 and c is even and
 c and Tablesize are coprime (eg by making Tablesize prime)
```

(2)
Are we sure of the average number of entries and maximum number.
If no: The closed table may have to change size dynamically incurring
and occasion large penalty. Does this matter. Open table more predictable.

Is deleting allowed. Easy with open table. Not so good with closed.

Does the application already use a general purpose space allocator, if
so this could be used for the hash chain nodes, and in a typical compiler.

(3a)
The average length of chain is 1.
Prob that a chain is of length 0 is (1-1/N)**N approx = 1/e (or 1/3)
so average length of non empty chains is about 1.5.

Expected number of comparisons to find an existing item in a chain of length n
is: (1+2+...n)/n = (n+1)/2

The average chain length is 1.5, so the expected number of comparisons is
(1.5+1)/2 = 1.25 (about)

(3b)
The average length of chain is 1/2 (plus a little because of clustering).
Prob that a chain is of length 0 is (1-1/2N)**N approx = e**-0.5 (or 1/1.7)
Let l = average length of non empty chains.
(1-1/1.7)l = 1/2   => l = (1/2) x 1.7/0.7 = 1.2 (approx)

The average chain length is 1.5, so the expected number of comparisons is
(1.2+1)/2 = 1.1 (about)