Fall 2009 CSE Qualifying Exam
Core Subjects

September 19, 2009
1. It’s 1997 and you’re a graduate student at Stanford named Larry Page. You’re trying to build a new Internet search engine and your strategy is to optimize its performance by ensuring that during a search, neither the CPU nor its disk array is idle.

The search database is logically divided into 100 MB contiguous blocks. After the first block is read, the engine reads subsequent blocks while using the CPU to search the previously read block. It takes 100 ms for the CPU to search each block.

You decide to use disks that each rotates at 170 revolutions/sec, has an average seek time of 8 ms, has a transfer rate of 50 MB/sec, and has a controller overhead of 2 ms. How many disks do you need in your disk array?

2. Determine the total branch penalty for a branch-target buffer assuming the penalty cycles for individual mispredictions shown below:

<table>
<thead>
<tr>
<th>Instruction in buffer</th>
<th>Prediction</th>
<th>Actual Branch</th>
<th>Penalty cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>yes</td>
<td>taken</td>
<td>taken</td>
<td>0</td>
</tr>
<tr>
<td>yes</td>
<td>taken</td>
<td>not taken</td>
<td>2</td>
</tr>
<tr>
<td>no</td>
<td>taken</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>no</td>
<td>not taken</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Make the following assumptions about the prediction accuracy and hit rate:

- Prediction accuracy is 90% (for instructions in the buffer)
- Hit rate in the buffer is 90% (for branches predicted taken)

3. You are building a system around a processor with in-order execution that runs at 4 GHz and has a CPI of 0.7 excluding memory accesses. The only instructions that read or write data from memory are loads (20% of all instructions) and stores (5% of all instructions).

The memory system for this computer is composed of a split L1 cache that imposes no penalty on hits. Both the I-cache and D-cache are direct-mapped and hold 64 KB each. The I-cache has a 1% miss rate and 64-byte blocks and the D-cache is write-through with a 7% miss rate and 16-byte blocks. There is a write buffer on the D-cache that eliminates stalls for 95% of all writes.

The 12 MB write-back, unified L2 cache has 64-byte blocks and an access time of 15 ns. It is connected to the L1 cache by a 128-bit data bus that runs at 266 MHz and can transfer one 128-bit word per bus cycle. Of all memory references sent to the L2 cache in this system, 80% are satisfied without going to main memory. Also, 50% of all blocks replaced are dirty.

The 128-bit-wide main memory has an access latency of 30 ns, after which any number of bus words may be transferred at the rate of one per cycle on the 128-bit-wide 133 MHz main memory bus.
(a) What is the average memory access time for instruction accesses?
(b) What is the average memory access time for data reads?
(c) What is the average memory access time for data writes?
(d) What is the overall CPI, including memory accesses?
Compilers

1. LR-Parsing

   (a) Give definitions of FIRST(α) and FOLLOW(X).

   (b) Consider the following grammar G:

   \[
   \begin{align*}
   S' & \rightarrow S \\
   S & \rightarrow V \ '=' \ E \\
   S & \rightarrow E \\
   S & \rightarrow V \\
   V & \rightarrow \text{id} \\
   V & \rightarrow ' \ast ' \ E \\
   E & \rightarrow V
   \end{align*}
   \]

   For the grammar G above generate six of the LR(1) sets of items.

   (c) Using the partial Sets-of-Items constructed in part (b) construct as much as you can of the parse table.

   (d) Describe in detail how the LR Parsing Algorithm will proceed when the state is \( s \), the next token is \( t \) and the stack contents is \( X_0, X_1, \ldots, X_{top1}, X_{top} \).

2. Middle exit loop

   Provide semantic actions for the middle break loop, which has the following syntax:

   \[
   S \rightarrow \text{loop } L_1 \text{ until } B \text{ } L_2 \text{ endloop ;}
   \]

   When the loop is entered, statement list \( L_1 \) is executed before the termination test \( B \). If \( B \) evaluates to true, then the loop exits. If not, execution proceeds with statement list \( L_2 \) and then with \( L_1 \) before testing \( B \) again.
3. The following fragment of intermediate code was generated by a compiler:

```
1    LabelA: i = 1000
2        sum = 0
3    LabelB: j = 1000
4        k = 1000
5    LabelC: t1 = j * 4000
6        t2 = i * 4
7        t3 = t1 + t2
8        t4 = a[t3]
9        if t4 < 0 goto LabelE
10       sum = sum + t4
11       goto LabelF
12    LabelD: k = k - 1
13        if k > 0 goto LabelC
14    LabelE: sum = sum - t4
15    LabelF: j = j - 1
16        if j > 0 goto LabelC
17        i = i - 1
18        if i > 0 goto LabelB
19    LabelG: no op
```

Assume that the only entry point is at LabelA.

(a) Decompose the code into basic blocks.
(b) Draw the control flow graph and identify any unreachable code if there is any.
(c) Describe the liveness of each variable just before each line in the block containing statement 9 assuming that only i, j, k, t4, and sum are live after 9.
(d) Describe any simplifying transformations that can be performed on the code (i.e., transformations that preserve the semantics but improve the code).
Algorithms

1. Suppose $T(n)$ is a positive-valued function defined for all positive integers $n$ and satisfying the recurrence

$$T(n) = T\left(\frac{2}{3}n\right) + T\left(\frac{3}{4}n\right) + n^2.$$ 

Using the tree method, show that $T(n) = \Omega(n^t)$ for some real constant $t > 2$. (Note: floors or ceilings in the recursive arguments don’t affect the asymptotics.) [Hint: You may find the following fact useful at some point: $a^{\log_b c} = c^{\log_b a}$ for any $a, b, c > 0$ such that $b \neq 1$.]

2. Describe Breadth First Search (BFS) precisely. Show that running BFS starting at a node $v \in V$ in a digraph $G = (V, E)$ visits nodes in ascending order of (unweighted) distance from $v$.

3. We will say that a Boolean formula $\varphi$ in conjunctive normal form is majority satisfiable if there is a truth assignment to the variables of $\varphi$ so that at least half of the literals in each clause are true. Show that the MAJORITY SATISFIABILITY problem (“Given $\varphi$ in CNF, is it majority satisfiable?”) is NP-hard by giving a polynomial reduction (Karp reduction) from the problem 3-CNFSAT. Is this problem in NP? Explain.
Theory

1. Let $x$ and $y$ be any two strings. A *merge* of $x$ and $y$ is any string obtained by merging the symbols of $x$ with those of $y$ in some arbitrary way, maintaining the order of the symbols from each string. More exactly, if $|x| = m$ and $|y| = n$, then a string $z = z_1 \cdots z_k$ is a merge of $x$ and $y$ if and only if

- $k = m + n$,
- there exist $1 \leq i_1 < i_2 < \cdots < i_m \leq k$ such that $x = z_{i_1} z_{i_2} \cdots z_{i_m}$,
- there exist $1 \leq j_1 < j_2 < \cdots < j_n \leq k$ such that $y = z_{j_1} z_{j_2} \cdots z_{j_n}$, and
- $\{i_1, \ldots, i_m\} \cap \{j_1, \ldots, j_n\} = \emptyset$.

For example, there are five different merges of the strings $ab$ and $bc$: $abbc$ $abcb$ $babc$ $bacb$ $bcab$

Let $A$ and $B$ be any languages over the same input alphabet $\Sigma$. Define $A \text{ merge } B := \{z \in \Sigma^* \mid z$ is a merge of some $x \in A$ and some $y \in B\}$.

Show that if $A$ and $B$ are regular, then so is $A \text{ merge } B$. [Hint: Given a DFA for $A$ with $r$ many states and an DFA for $B$ with $s$ many states, you can construct an NFA for $A \text{ merge } B$ with $rs$ many states.]

2. Let $A$ and $B$ be any Turing-recognizable languages. Show that there exist languages $\hat{A}$ and $\hat{B}$ such that

- $\hat{A}$ and $\hat{B}$ are both Turing-recognizable,
- $\hat{A} \subseteq A$,
- $\hat{B} \subseteq B$,
- $\hat{A} \cup \hat{B} = A \cup B$, and
- $\hat{A} \cap \hat{B} = \emptyset$.

(Notice that setting $\hat{A} := A$ and $\hat{B} := B - A$ satisfies the last four criteria above but not the first, because $B - A$ might not be Turing-recognizable.) [Hint: Take a TM for $A$ and a TM for $B$ and run them simultaneously on the same input.]

3. Show that if a Turing machine $M$ runs using space $O(n)$ on inputs of length $n$ and halts on all inputs, then $M$ runs in time $2^{O(n)}$. 

7