Comprehensive Exam: Algorithms and Concrete Mathematics Autumn 2007

This is a one hour closed-book exam and the point total for all questions is 60.

In questions that ask you to provide an algorithm, please explain the algorithm in words and diagrams, no need to write code or pseudo code. Also, for any algorithm, prove correctness and prove its running time. No credit will be given for exponential-time algorithms. Polynomial but slow algorithms will get some partial credit. Amount of credit will depend on how much slower they are compared to what is achievable using the knowledge in the reading list. Correct and fast algorithms with incomplete proof of correctness will get small number of points.

For full credit, the answers should be short and precise. Long and convoluted answers will not get full credit even if they are correct.

1. **[14 pts]** Please answer "true" or "false" to each one of the following questions. Correct answers will give you **(2 pts)** each while wrong answers will reduce your score by **(2 pts)** each.

   (a) There exists $\epsilon > 0$ such that $n^\epsilon = O(\log n)$.

   (b) Suppose $T(n)$ is given by the recurrence $T(n) = T(\lfloor n/\log n \rfloor) + n$; $T(1) = 1$. Then, $T(n) = \Theta(n)$.

   (c) Given a heap data structure (organized with minimum on top), it is possible to find the second-smallest element in constant time.

   (d) There exists a deterministic linear time algorithm that, given two English words with at most n characters each, determines whether they are anagrams of each other.

   (e) In comparison model, the lower bound on finding median of n input numbers is $\Omega(n \log n)$.

   (f) You are given a graph $G = (V, E)$ with positive lengths on edges and a shortest path P from $v \in V$ to $u \in V$. Next, lengths are transformed by computing square of the length of each edge. (If old edge length was w, new one is w^2). Claim: it is guaranteed that P is a shortest path from v to u with respect to new lengths as well.

   (g) Given a weighted graph and two nodes, it is possible to list all shortest paths between these two nodes in polynomial time.

2. **[18 pts]** You have a single classroom and a list of lectures that you would like to use the classroom for. Each lecture is specified by start time and end time. Your goal is to come up with a schedule that maximizes the number of lectures scheduled in the classroom under the constraint that no two scheduled lectures can overlap. For example, if one lecture needs the 1pm to 2pm slot and the other one needs 1:30pm to 3pm slot, at most one of these lectures can be scheduled. Design an efficient algorithm to solve the problem. Prove correctness and running time.

3. **[12 pts]** Prove that if edge weights of a graph are unique (no two edges have the same weight), then there is unique solution to the minimum-cost spanning tree problem.

4. **[16 pts]** Given an acyclic directed graph $G = (V, E)$ and a node $s \in V$, describe an algorithm to find the number of paths from s to each one of the nodes in V.

Comprehensive Exam: Algorithms and Concrete Mathematics Autumn 2007

This is a one hour closed-book exam and the point total for all questions is 60.

In questions that ask you to provide an algorithm, please explain the algorithm in words and diagrams, no need to write code or pseudo code. Also, for any algorithm, prove correctness and prove its running time. No credit will be given for exponential-time algorithms. Polynomial but slow algorithms will get some partial credit. Amount of credit will depend on how much slower they are compared to what is achievable using the knowledge in the reading list. Correct and fast algorithms with incomplete proof of correctness will get small number of points.

For full credit, the answers should be short and precise. Long and convoluted answers will not get full credit even if they are correct.

1. **[14 pts]** Please answer "true" or "false" to each one of the following questions. Correct answers will give you **(2 pts)** each while wrong answers will reduce your score by **(2 pts)** each.

   (a) There exists $\epsilon > 0$ such that $n^\epsilon = O(\log n)$.

   Answer: FALSE

   (b) Suppose $T(n)$ is given by the recurrence $T(n) = T(\lfloor n/\log n \rfloor) + n$; $T(1) = 1$. Then, $T(n) = \Theta(n)$.

   Answer: TRUE. Lower bound is obvious. Upper bound follows from the fact that $\lfloor n/\log n \rfloor$ is bounded by $n/2$ for large enough n.

   (c) Given a heap data structure (organized with minimum on top), it is possible to find the second-smallest element in constant time.

   Answer: TRUE

   (d) There exists a deterministic linear time algorithm that, given two English words with at most n characters each, determines whether they are anagrams of each other.

   Answer: TRUE. Count the frequency of all 26 letters and compare.

   (e) In comparison model, the lower bound on finding median of n input numbers is $\Omega(n \log n)$.

   Answer: FALSE. Median can be computed in linear time.

   (f) You are given a graph $G = (V, E)$ with positive lengths on edges and a shortest path P from $v \in V$ to $u \in V$. Next, lengths are transformed by computing square of the length of each edge. (If old edge length was w, new one is w^2). Claim: it is guaranteed that P is a shortest path from v to u with respect to new lengths as well.

   Answer: FALSE. Example: graph with three nodes, $w(ab) = w(bc) = 1$, and $w(ac) = 2$. Edge ac is shortest path between a and c in the original graph. After squaring of the costs this is not true anymore.

   (g) Given a weighted graph and two nodes, it is possible to list all shortest paths between these two nodes in polynomial time.

   Answer: FALSE. There can be an exponential number of shortest paths.

2. **[18 pts]** You have a single classroom and a list of lectures that you would like to use the classroom for. Each lecture is specified by start time and end time. Your goal is to come up with a schedule that maximizes the number of lectures scheduled in the classroom under the constraint that no two scheduled lectures can overlap. For example, if one lecture needs the 1pm to 2pm slot and the other one needs 1:30pm to 3pm slot, at most one of these lectures can be scheduled. Design an efficient algorithm to solve the problem. Prove correctness and running time.

   Answer: Sketch of the algorithm. Greedy approach: sort by finish time; pick the first lecture, allocate it, delete overlapping lectures, repeat. Claim: there exists an optimum

solution that includes the first lecture we allocated. Proof: Take any optimum solution and replace the first allocated lecture in this solution with the first lecture in our solution. Our lecture finishes no later than the first allocated lecture in the OPT, thus no overlaps created, and thus the modified solution is legal. This solution has the same number of lectures scheduled and includes the first lecture in our solution. QED. The claim implies that we can greedily commit to the earliest-finish-time lecture. Once the lectures are sorted, each iteration (picking the next lecture and deciding whether we schedule it or drop it) takes constant time. Total running time is $O(n \log n)$.

3. **[12 pts]** Prove that if edge weights of a graph are unique (no two edges have the same weight), then there is unique solution to the minimum-cost spanning tree problem.

   Answer: Let T be the tree produced by Kruskal's algorithm and let T' be a different minimum-cost tree. Let uv be the first edge where Kruskal's algorithm disagrees with T', i.e. this is the smallest-weight edge that Kruskal's algorithm put into T but uv is not part of T'. [Observe that this is the only possible disagreement since if uv was rejected by Kruskal's algorithm, it closes a cycle together with already chosen edges. But all these edges are in T' as well.] Add uv to T', creating a cycle. Note that the path from u to v in T' has to include at least one edge with higher weight than $w(uv)$ since when uv was chosen by Kruskal's algorithm, there was no path from u to v using already chosen edges. Cut-and-paste argument finishes the proof.

4. **[16 pts]** Given an acyclic directed graph $G = (V, E)$ and a node $s \in V$, describe an algorithm to find the number of paths from s to each one of the nodes in V.

   Answer: First compute topological sort. Then use a simple dynamic programming. For any vertex v, let $P[v]$ denote the number of paths from s to v. Initialize $P[v] = 0$ for all $v \in V$, and set $P[s] = 1$. Compute the recurrence $P[v] = \sum_{uv \in E} P[u]$ in topological order.

   Topological sort guarantees that $P[v]$ depends only on values of $P[]$ for nodes that are before v in topological sort. Thus, when we compute $P[v]$, the $P[]$ values for all these nodes were already computed. Topological sort can be implemented to run in $O(|E|+|V|)$. Second phase of the algorithm scans nodes one-by-one, each time examining all incoming edges, again giving $O(|E| + |V|)$.

2007 Comprehensive Examination
Artificial Intelligence

1. Search. (20 points) Assume we have a binary search tree with a maximum depth at level *d*. You can assume that the root node is level 0. Just give answers; no explanation necessary.

a. If there is a solution at depth $k<d$, what is the maximal number of nodes that will be examined by depth-first search, breadth-first search, and by iterative deepening (with depth starting at 0 and incrementing by 1 on each iteration)?

b. If there is a solution at depth $k<d$, what is the maximal amount of storage (in terms of nodes) required by depth-first search, breadth-first search, and by iterative deepening?

c. Suppose the tree is an adversary search tree with depth $d=2n$ and evaluation at depth *d*, what is a good asymptotic bound on the minimum number of nodes that must be searched with alpha-beta pruning?

d. What is the maximum amount of storage (in terms of nodes) required for alpha-beta pruning in this minimal case?

2. Constraint Satisfaction Problems. (20 points) Consider the following five algorithms for solving CSPs.

(a) Depth-first search with consistency checking and a fixed variable ordering.

(b) Depth-first search with forward checking and a fixed variable ordering.

(c) Depth-first search with forward checking and the most-constrained variable heuristic.

(d) Depth-first search with AC-3 and a fixed variable ordering.

(e) Depth-first search with AC-3 and the least-constrained variable heuristic.

Write down all pairs of algorithms where the first is better than the second, i.e. write down "$x \leq y$" if and only if algorithm y is *guaranteed* to expand at least as many nodes of the search tree as x. No explanation necessary.

3. Validity, Contingency, Unsatisfiability. (20 points) For each of the following sentences, say whether it is valid, unsatisfiable, or contingent (neither valid nor unsatisfiable). You do not need to justify your answers.

(a) $\forall x.p(x) \Rightarrow p(x)$

(b) $p(x) \Rightarrow \forall x.p(x)$

(c) $\exists x.p(x) \Rightarrow p(x)$

(d) $p(x) \Rightarrow \exists x.p(x)$

(e) $\exists x.p(x) \Rightarrow \forall x.p(x)$

(f) $\forall x.p(x) \Rightarrow \exists x.p(x)$

(g) $\forall x.p(x) \Rightarrow \exists x.\neg p(x)$

(h) $\forall x.(p(x) \Rightarrow q(x)) \Rightarrow \exists x.(p(x) \wedge q(x))$

(i) $\forall x.(p(x) \Rightarrow q(x)) \wedge \neg\exists x.(p(x) \wedge q(x))$

(j) $(\exists x.p(x) \Rightarrow \forall x.p(x)) \vee (\forall x.p(x) \Rightarrow \forall x.q(x))$

4. Resolution. (20 points) Use the resolution method and the following premises to prove the conclusion shown below.

Premises:

a. $\forall x.\forall y.(p(x, y) \Rightarrow \exists z.q(x, y, z))$

b. $\forall x.(\exists y.r(x, y) \Rightarrow \neg\exists w.\exists z.q(x, w, z))$

c. $\forall x.\exists y.(p(x, y) \vee \forall z.q(x, y, z))$

Conclusion:

$\forall x.\exists y.\exists z.(\neg r(y, z) \wedge q(x, y, z))$

Note that this is a question about Resolution. You will get zero points (nil, nada, rien, zip, nothing) unless you prove it using the standard resolution procedure.

5. Machine Learning. (20 points)

(a) Consider a generative model for nonnegative real-valued inputs x where each x is sampled from the uniform distribution over the real interval $[0, \theta]$ and where θ is a real-valued parameter. Given a training set $\{x^1, x^2, \ldots, x^n\}$ with each x^i is sampled i.i.d. from this generative distribution, give the maximum likelihood estimate for θ.

(b) For a binary classification task, suppose you are given a training set with n examples, and also a very large test set for evaluation. You train a logistic regression classifier for this task with the parameters estimated to maximize likelihood on the training set. For the trained classifier, let the misclassification error on the training set be e_{train} and the misclassification error on the test set be e_{test}, and suppose the test set error e_{test} is unacceptably high. You are given two choices: (A) Ask for more training data, or (B) Try a richer, non-linear classifier (e.g., try an SVM with the RBF kernel). For each of the following scenarios, which of the two choices should you investigate first. Give very, very, very brief explanations for your answers.

(i) $0 \approx e_{\text{train}} << e_{\text{test}}$

(ii) $e_{\text{train}} \approx e_{\text{test}} >> 0$

2007 Comprehensive Examination Solutions
Artificial Intelligence

1. Search. (20 points)

(a) Depth-first: $k + 1$
Breadth-first: 2^k
Iterative Deepening: $2^{k+1} - 1$

(b) Depth-first: $2^{d+1} - 2^{d-k+1} + 1$
Breadth-first: $2^{k+1} - 1$
Iterative Deepening: $\sum_{i=0}^{k}(2^{i+1} - 1) =$

$$(2^{k+1} - 1) + \ldots + (2^{0+1} - 1) =$$
$$2^{k+1} + \ldots + 2^1 - k - 1 =$$
$$2^{k+2} - k - 3$$

(c) Depth-first: $d + 1$
Breadth-first: 2^k
Iterative Deepening: $k + 1$

2. Constraint Satisfaction Problems. (20 points) (d) ≤ (b) ≤ (a).

3. Logic. (20 points)

(a) Valid
(b) Contingent
(c) Contingent
(d) Valid
(e) Contingent
(f) Valid
(g) Contingent
(h) Contingent
(i) Unsatisfiable
(j) Valid

4. Resolution. (20 points)

1. $\{\neg p(x, y), q(x, y, f(x,y))\}$	Premise
2. $\{\neg r(x,y), \neg q(x, w, z)\}$	Premise
3. $\{p(x, g(x)), q(x, g(x), z)\}$	Premise
4. $\{r(y, z), \neg q(a, y, z)\}$	Negated Goal
5. $\{\neg q(a, x, y), \neg q(x, w, z)\}$	2, 4
6. $\{q(x, g(x),f(x, g(x))), q(x, g(x), z)\}$	1, 3
7. $\{\neg q(g(a), w, z)\}$	5, 6 (factoring 6)
8. $\{\}$	6, 7 (factoring 6)

5. Machine Learning. (20 points)

(a) The maximum likelihood estimate is: $\max_i x^i$

(b)(i) Try choice A. The current algorithm appears to be overfitting the training set. High variance problem.

(b)(ii) Try choice B. The current model cannot represent even the training data well. High bias problem.

Automata and Formal Languages Comprehensive Exam (60 Points)

Fall 2007

Note: this 1-hour exam is *closed book*.

Problem 1 (18 points)

(a) Give an algorithm that takes a DFA over the alphabet $\{0, 1\}$ as input and decides whether or not it accepts a non-empty language. Your algorithm should run in time linear in the number of states of the DFA.

(b) Recall that in a CNF grammar, every production either has the form $A \to BC$ (where A, B, C are variables) or $A \to a$ (where A is a variable and a is a terminal).

Fix a CNF grammar G. Give an algorithm that takes a string w as input and decides whether or not w is a member of the language generated by G. Your algorithm should run in time polynomial in the length n of the given string w (e.g., $O(n^3)$ time is fine).

Problem 2 (12 points)

Classify each of the following languages as being in one of the following classes of languages: *empty, finite, regular, context-free, recursive, recursively enumerable, all languages*. You must give the *smallest* class that contains *every possible language* fitting the following definitions. For example, the language of a DFA must always be context-free, but the smallest class that contains all such languages is that of the *regular* languages. *Do not provide explanations. Correct answers receive 3 points, incorrect answers receive -2 points.*

(a) $\{0^n 1^n 2^n \mid n \geq 1\}$.

(b) $\{a^{n!} \mid n \geq 1\}$.

(c) An NP-complete problem.

(d) The intersection of two context-free languages.

Problem 3 (12 points)

Decide whether each of the following are recursive, RE-but-not-recursive, or non-RE. *Do not provide explanations. Correct answers receive 3 points, incorrect answers receive -2 points.*

(a) The set of all TM codes for TMs that halt on every input.

(b) The set of all TM codes for TMs that halt on no input.

(c) The set of all TM codes for TMs that halt on at least one input.

(d) The set of all TM codes for TMs that, on at least one input, fail to halt.

Problem 4 (18 points)

Assume that the following *Node-Cover problem* is NP-complete: given an undirected graph G and an upper limit k, does there exist a node cover (a set of nodes of G that contains at least one endpoint of each edge of G) with at most k nodes?

Use this to prove that the following *Clique* problem is NP-complete: given an undirected graph G and a lower limit k, does there exist a clique (a set of nodes of G such that there is an edge between each pair of nodes in the set) with at least k nodes?

Automata and Formal Languages Comprehensive Exam Solutions

Fall 2007

Problem 1 (18 points)

(a) See Section 4.3.2 of HMU (pages 151–152 of the 2nd edition).

(b) See Section 7.4.4 of HMU (pages 299–301 of the 2nd edition).

Problem 2 (12 points)

Classify each of the following languages as being in one of the following classes of languages: *empty, finite, regular, context-free, recursive, recursively enumerable, all languages.*

(a) Recursive.

(b) Recursive.

(c) Recursive.

(d) Recursive.

Problem 3 (12 points)

(a) Not RE.

(b) Not RE.

(c) RE but not recursive.

(d) Not RE.

Problem 4 (18 points)

See the Web solution to Exercise 10.4.1(c) of HMU.

Compilers Comprehensive, November 7, 2007

1. (10 points)

   Suppose a lexical analyzer generator such as lex, flex, jlex, or jflex, were given the series of patterns:

   ab
   abc
   (ab?c)*
   . (“.” matches any individual character except newline)

   What strings would be recognized in the input `abcaccaabbacac` by repeated calls to the lexer?

   Briefly explain why the lexer would do this.

2. (20 points)

   The following is the relevant fragment of a YACC (actually, Bison) grammar for simple Boolean formulas. This (admittedly odd) grammar translates Boolean expressions to “reverse Polish” notation: `a | b & c` is translated to “a b OR c AND.” But, it also transforms the expression to negation normal form, using De Morgan’s laws to “push NOTs down to the leaves of the formula,” and it does this *on-the-fly without using any additional data structures or variables.* The negation normal form of `~(a | ~(b & c) | ~d)` is `~a & b & c & d)`.

```
%union yystacktype {char *name;  int flag; }

%start Formula

%token <name> ID
%left '|'
%left '&'
%nonassoc '~'

%%

F : F '&' { $<flag>$ = $<flag>0; } F
    { if ($<flag>0) {printf("OR\n"); } else { printf("AND\n"); }}
  | F '|' { $<flag>$ = $<flag>0; } F
    { if ($<flag>0) {printf("AND\n"); } else { printf("OR\n"); }}
  | '~' { $<flag>$ = !$<flag>0; } F
  | ID { printf("%s\n", $1); if ($<flag>0) { printf("NOT\n"); }}
  | '(' { $<flag>$ = $<flag>0; } F ')'
  ;
```

   (a) If the `%left` and `%nonassoc` declarations are omitted, bison reports “6 shift/reduce conflicts.”

      i. What is a shift/reduce conflict in an LALR parser?

      ii. Describe one specific conflict that occurs in this example.

iii. The YACC input above does not produce any errors about conflicts, because they are all resolved. Explain why this happens and how it works.

(b) This grammar has actions embedded in the middle of the productions. In YACC-style parser generators, this is "syntactic sugar" for a grammar with more rules, but no embedded actions (actions appear only at the right-hand ends of productions, to be executed when those productions are reduced). Please explain this how to de-sugar a grammar like this into one where all actions occur when productions are reduced.

(c) Basically, how does the YACC grammar with actions work? How does the parser keep track of whether the next sub-expression is in a negated context, especially when it exits the scope of a negation (as when it finishes parsing `~(b & c)` in the above example)?

3. (15 points)

   Describe an implementation for a symbol table supporting the following operations. You should assume that, for practical purposes, hash table insertions and lookups require constant time. If your solution does not meet the performance goals below, it will still receive some credit so long as it is clearly correct. You may assume that a symbol is bound only once in each scope.

   - `pushscope()` – mark a new declaration scope, e.g., at the beginning of a function declaration. This should be a constant-time operation.
   - `popscope()` – restore the symbol table to its state just before the matching pushscope. This should take $O(k)$ time, where k is the number of symbols declared in the scope that is popped.
   - `declare(symbol, decl)` – Associate "symbol" with "decl" in the current scope. decl is a datastructure describing, for example, a variable or a type declaration for the symbol. This should take constant time (not counting the time to construct the decl).
   - `lookup(symbol)` – Find and return the decl most recently associated with symbol in a scope that has not been popped. This should take constant time.

4. (15 points) How does compile-time function (or method) overloading interact with type-checking in programming language implementations? Briefly discuss how type evaluation could work in two scenarios: when overloaded functions are resolved based only on the types of the arguments of the functions, and when the return type of overloaded functions is used to resolve overloading, in addition to the argument types. Don't worry about automatic type conversions (e.g., promotion from integers to floats) or inheritance.

Solutions to Compilers Comprehensive, November 7, 2007

1. (10 points)

   Suppose a lexical analyzer generator such as lex, flex, jlex, or jflex, were given the series of patterns:

   ab
   abc
   (ab?c)*
   . (“.” matches any individual character except newline)

   What strings would be recognized in the input `abcaccaabbacac` by repeated calls to the lexer?

   Briefly explain why the lexer would do this.

   Solution: `abcac|c|a|ab|b|acac`

   On each call, the lexer matches the longest string it can against its patterns. For example, `a`, `ab`, `abc`, etc. all match the first part of the input, but `abcac` is the longest matching prefix.

2. (20 points)

   The following is the relevant fragment of a YACC (actually, Bison) grammar for simple Boolean formulas. This (admittedly odd) grammar translates Boolean expressions to “reverse Polish” notation: `a | b & c` is translated to “a b OR c AND.” But, it also transforms the expression to negation normal form, using De Morgan’s laws to “push NOTs down to the leaves of the formula,” and it does this *on-the-fly without using any additional data structures or variables.* The negation normal form of `~(a | ~(b & c) | ~d)` is `~a & b & c & d)`.

```
%union yystacktype {char *name;  int flag; }

%start Formula

%token <name> ID
%left '|'
%left '&'
%nonassoc '~'

%%

F : F '&' { $<flag>$ = $<flag>0; } F
    { if ($<flag>0) {printf("OR\n"); } else { printf("AND\n"); }}
  | F '|' { $<flag>$ = $<flag>0; } F
    { if ($<flag>0) {printf("AND\n"); } else { printf("OR\n"); }}
  | '~' { $<flag>$ = !$<flag>0; } F
  | ID { printf("%s\n", $1); if ($<flag>0) { printf("NOT\n"); }}
  | '(' { $<flag>$ = $<flag>0; } F ')'
  ;
```

   (a) If the `%left` and `%nonassoc` declarations are omitted, bison reports “6 shift/reduce conflicts.”

i. What is a shift/reduce conflict in an LALR parser?
Solution: It occurs in a parser state that have both shift and reduce items, where the lookahead symbols for the reduce item include the next terminal symbol in a shift item. E.g., the state could have items $[A \rightarrow \alpha \cdot a\beta, X]$ and $[B \rightarrow \gamma, a]$.
The parser must commit to a shift or reduce action without sufficient context to determine which will lead to a successful parse.
In this case, the problem is that the grammar is ambiguous, so both parses will be successful, but lead to different translations for the logical expression.

ii. Describe one specific conflict that occurs in this example. **Solution:**
Here are all of them:

```
F -> F & F . vs F -> F . & F. (left or right associative?)
F -> F | F . vs F -> F . | F. (left or right associative?)
F -> F | F . vs F -> F . & F. (| vs & precedence)
F -> F & F . vs F -> F . | F. (& vs | precedence)
F -> ~ F . vs F -> F . | F.   (~ vs | precedence)
F -> ~ F . vs F -> F . & F.   (~ vs & precedence)
```

iii. The YACC input above does not produce any errors about conflicts, because they are all resolved. Explain why this happens and how it works.
Solution: The `%left` and `%nonassoc` declarations define whether `&` and `|` are left or right associative, and the order of the declarations gives the relative precedence of the operators (lower precedence first). Bison/YACC/etc. resolve these conflicts by using the precedence rules to choose which action goes into the ACTION table entry.

(b) This grammar has actions embedded in the middle of the productions. In YACC-style parser generators, this is "syntactic sugar" for a grammar with more rules, but no embedded actions (actions appear only at the right-hand ends of productions, to be executed when those productions are reduced). Please explain this how to de-sugar a grammar like this into one where all actions occur when productions are reduced.

Solution: For each action, a new non-terminal symbol is generated (say A_i and a new production is added (e.g., $A_i \rightarrow \epsilon$). The embedded action is performed when this new production is reduced. YACC goes to some special effort to make `$i` symbols in the action of this refer to the proper positions in the value stack, based on the positions of the corresponding symbols in the original rule, not the newly introduced rule.

(c) Basically, how does the YACC grammar with actions work? How does the parser keep track of whether the next sub-expression is in a negated context, especially when it exits the scope of a negation (as when it finishes parsing `~ (b & c)` in the above example)?

Solution: Whenever the grammar is about to parse a `F`, it makes sure that the value on top of the value stack is 1 iff `F` is in a negated context (it copies the flag as necessary). When parsing an `F`, this is `$0`, which refers to the position on the value stack just before `$1`, i.e. just under the position for the leftmost symbol of the RHS of the current production. When the parser reduces a `~` formula, the flag saying whether the current context is negated, leaving the flag for the next outer context in the right position on the stack for the next formula.

3. (15 points)

Describe an implementation for a symbol table supporting the following operations. You should assume that, for practical purposes, hash table insertions and lookups require constant time. If your

solution does not meet the performance goals below, it will still receive some credit so long as it is clearly correct. You may assume that a symbol is bound only once in each scope.

- `pushscope()` – mark a new declaration scope, e.g., at the beginning of a function declaration. This should be a constant-time operation.
- `popscope()` – restore the symbol table to its state just before the matching pushscope. This should take $O(k)$ time, where k is the number of symbols declared in the scope that is popped.
- `declare(symbol, decl)` – Associate "symbol" with "decl" in the current scope. decl is a datastructure describing, for example, a variable or a type declaration for the symbol. This should take constant time (not counting the time to construct the decl).
- `lookup(symbol)` – Find and return the decl most recently associated with symbol in a scope that has not been popped. This should take constant time.

Solution: We assume that symbols are strings or records with strings an other data, and that decls are pointers to records representing decls. A symbol table entry is a record with a pointer to a symbol, a pointer to a decl, a "saved" link to another entry (or null) and a "next_in_scope" link to another decl. There is a hash table with symbols as keys and entries as values, and a separate "scopes" stack of pointers to entries.

`pushscope()`: pushes a null onto the scope stack.

`declare(symbol, decl)`: Allocate a new entry, set the symbol and decl fields to symbol and decl, set next_in_scope to the top entry in the scope stack, overwrite the top of the scope stack with the new entry, set the ''saved" field to the current value of key in the hash table (or null if there is no value), store key $\rightarrow$ symbol in the hash table.

`lookup(symbol) -> decl`: Lookup symbol in hash table. Report an error if there is no value. Otherwise, return the contents of the decl field in the entry that is found.

`popscope()`: Get entry on top of scope stack. Iterate over the linked list of entries created by the "next_in_scope" fields, removing each entry from the hash table using the "symbol" field and storing the "saved" value of the entry under the key (if non-null) Pop the scope stack. (Note: A nice implementation of a hash table might provide a way to update or delete the entry directly without hashing multiple times).

4. (15 points) How does compile-time function (or method) overloading interact with type-checking in programming language implementations? Briefly discuss how type evaluation could work in two scenarios: when overloaded functions are resolved based only on the types of the arguments of the functions, and when the return type of overloaded functions is used to resolve overloading, in addition to the argument types. Don't worry about automatic type conversions (e.g., promotion from integers to floats) or inheritance.

   Solution: Without overloading, type checking usually involves assigning types to expressions bottom-up: given the types of the children of an operator or function call, the result type can be determined. Type errors are detected when the children of the operator or function have inappropriate types.

   If function overloading depends only on the function arguments (children), types can still be assigned by a bottom-up pass. Once the types of the arguments to a function call have been determined, the compiler can find the correct overloaded function definition that matches those types. If no such

function exists, it is a type error. If there are multiple matching functions, it should also be an error. Otherwise, there is a unique function, and the return type of that function is the type of the current expression.

If the function return type can be used to resolve ambiguities, the match of the correct function depends on what type the surrounding context "wants." E.g., the function call is an argument to another function call, which needs a particular type, or is assigned to a variable of a particular type. If only one of the overloaded function definitions results in a correctly typed expression, that is the correct function to choose. If there are no functions or multiple functions, it is an error. In this case, assigning types involves both top-down and bottom-up passes – and may need to be repeated to propagate type constraints over the abstract syntax tree.

Computer Architecture Comprehensive Exam

Exam Instructions
Answer each of the questions included in the exam. Write all of your answers directly on the examination paper, including any work that you wish to be considered for partial credit. The examination is open-book, and you may make use of the text, handouts, your own course notes, and a calculator.

On equations: Wherever possible, make sure to include the equation, the equation rewritten with the numerical values, and the final solution. Partial credit will be weighted appropriately for each component of the problem, and providing more information improves the likelihood that partial credit can be awarded.

On writing code: Unless otherwise stated, you are free to use any of the assembly instructions listed in the Appendix at the back of the book, including pseudoinstructions. You do not need to optimize your MIPS code unless specifically instructed to do so.

On time: You will have one hour to complete this exam. Budget your time and try to leave some time at the end to go over your work. The point weightings correspond roughly to the time each problem is expected to take.

THE STANFORD UNIVERSITY HONOR CODE

The Honor Code is an undertaking of the students, individually and collectively:

1. that they will not give or receive aid in examinations; that they will not give or receive unpermitted aid in class work, in the preparation of reports, or in any other work that is to be used by the instructor as the basis of grading;
2. that they will do their share and take an active part in seeing to it that others as well as themselves uphold the spirit and letter of the Honor Code.

I acknowledge and accept the Honor Code.

Magic Number: _____

		Score	*Grader*
1. Short Answer	(15)	_____	_____
2. Pipelining	(15)	_____	_____
3. Memory Heirarchy	(15)	_____	_____
4. Cache Math	(5)	_____	_____
5. MIPS Assembly	(10)	_____	_____
TOTAL	(60)	_____	

Problem 1: Short Answer (15 points)

Please provide short, concise answers.

(1) *(2 points)* Your company, Acme Corp., is deciding between two computer systems to deploy its new killer Road Runner Tracking application. Ben Bitdiddle says that his company's system has the better performance because it has the higher clock speed and the higher IPC. Explain why his logic is flawed.

(2) *(2 points)* Some RISC architectures require the compiler (or assembly programmer) to guarantee that a register not be accessed for a given number of cycles after it is loaded from memory. Give an advantage and a disadvantage of this design choice.

(3) *(2 points)* Ben Bitdiddle is writing an optimizing compiler for an architecture that supports virtual memory. He notices that his target processor can execute an unaligned 32 bit load more quickly than an 8 bit load. Is it okay for his compiler to emit 32 bit loads and then ignore the extra 24 bits? Why or why not?

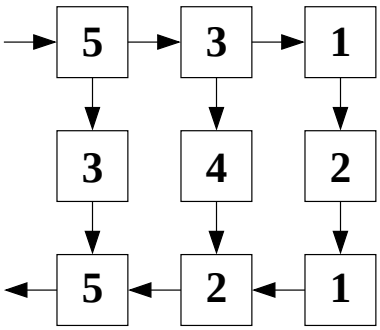
(4) *(3 points)* Briefly describe the data access pattern of an application for which an LRU (least recently used) cache replacement policy performs worse than a random replacement policy.

(5) *(3 points)* Briefly describe the data access pattern of an application for which a cache with a random replacement policy performs worse than an LRU replacement policy.

(6) *(3 points)* Briefly give two ways in which loop unrolling can increase performance and one in which it can decrease performance.

Problem 2: Pipelining (15 points)

The National Security Agency has designed a new encryption device called the Conundrum, composed of nine combinational modules connected as shown in the diagram below:

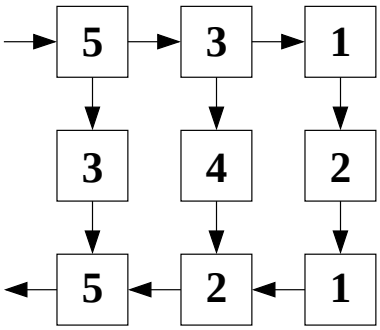


The device takes an integer value X and computes an encrypted version C(X). In the diagram above each combinational component is marked with its propagation delay in microseconds; contamination delays are zero for each component. Assume an ideal (zero-delay) register on both the input and the output of the device.

(1) *(4 points)* What is the latency and throughput of the Conundrum device?

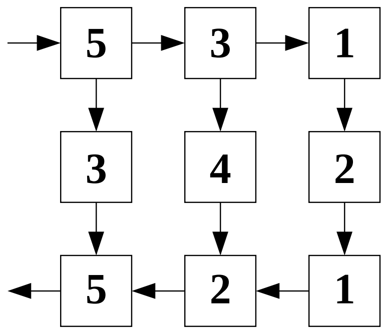
latency (μs) _____ throughput ($1/\mu s$) _____

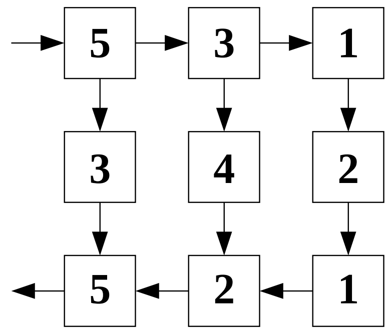
(2) *(9 points)* The NSA needs to produce a version of the Conundrum device that has a throughput larger than 1/15 but wants the implementation with smallest latency (cycle time * pipeline depth) that meets the throughput constraint. Using the diagram below indicate the locations of ideal (zero-delay) registers to create a pipelined implementation that meets these goals. You may use as many registers as necessary. We'll give partial credit for designs with throughput larger than 1/15; full credit for achieving the smallest possible latency. (Extra copies of this diagram can be found on the following page, but only solutions written on the diagram below will be graded).

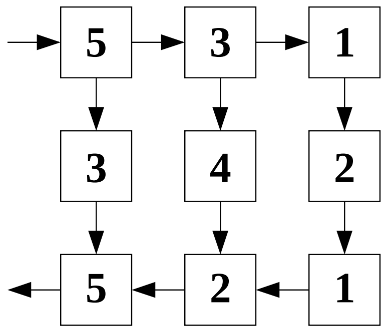


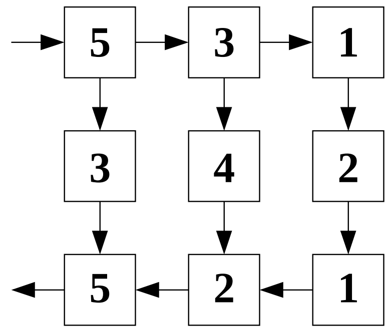
(3) *(2 points)* What is the latency of your pipelined implementation?

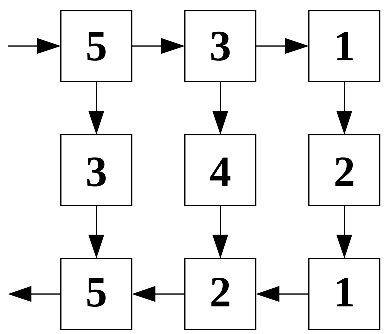
latency (μs) _____ throughput ($1/\mu s$) _____

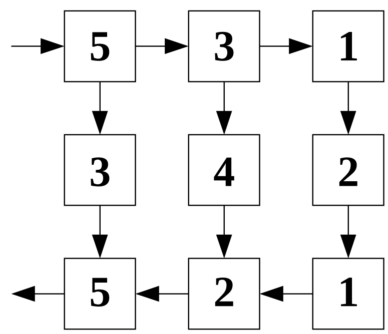












(These diagrams are for scratch work; no solution written here will be graded. Record your solution on the previous page.)

Problem 3: Memory Heirarchy (15 points)

Assume you have a 1 GHz processor with 2-levels of cache and DRAM main memory. The first level cache is split for instructions and data. The system does not use early restart or critical word first, i.e. data blocks must be completely transfered before their results are available. The memory system has the following parameters (Note that here 1KB = 1024 bytes):

	Hit Time	Miss Rate	Block Size
Level-1 cache	1 cycle	6% for data 2% for instruction	32 bytes
Level-2 cache	12 cycles + (1 cycle per 64 bits)	2%	256 bytes
DRAM	70ns + (10ns per 8 bytes)	–	–

The system includes a TLB with a miss rate of 0.5% for data and never incurs a TLB miss for instructions. The TLB miss penalty is 300 cycles and TLB hits take place in parallel with level-1 cache access. All caches in the system are virtually indexed and physically tagged. Assume that the system never swaps memory out to disk.

(1) *(5 points)* What is the average memory access time (AMAT) in clock cycles for instructions?

(2) *(5 points)* What is the AMAT in clock cycles for data? Assume all data accesses are loads.

(3) *(5 points)* Suppose that we measure the following instruction mix for a program:

Loads: 25%, Stores: 15%, Integer: 30%, Floating-Point: 20%, Branches: 10%

Assume that the processor is using the 5-stage pipeline (base CPI of 1.0). Data hazards cause an average penalty of 0.9 cycles for floating point operations. Integer operations run at maximum throughput. The processor uses the predict-branch-not-taken technique, which turns out to be correct for 80% of the branches. For the remaining branches, there is a 1 cycle stall. What is the average CPI of this program including memory misses (from questions a and b)? Assume that stores have the same AMAT as loads.

Problem 4: Cache Math (5 points)

Answers to this question may be in the form of a bare number (decimal or hex), 2^n, or nk (= $n*2^{10} = n*1024$), for example, 65536, 0x10000, 2^{16}, or 64k are all equivalent and acceptable answers.

Consider a 256kb 4-way set associative cache with 256 byte cache lines for a processor that uses 64-bit data words and 48-bit byte addresses. Assume the variable x, of type uint64_t, is stored in memory at location 0x4A85_B413_A518.

(1) *(2 points)* Fill in the bit ranges in the following diagram. Bit ranges should be inclusive. For example, if a field uses bits 0, 1, 2, and 3, label it 3:0.

47 :	:	:	: 0
tag	set	word in line	byte in word

(2) *(3 points)* Assume that x is present in the cache, and char* ptr = 0x4A85_B400_0000. Determine if the following accesses will cause a cache miss. For each access circle MISS if it must cause a miss, HIT if it will never cause a miss, or NOT ENOUGH INFO.

*(ptr + 0x11_A538)	MISS	HIT	NOT ENOUGH INFO
*(ptr + 0x13_A588)	MISS	HIT	NOT ENOUGH INFO
*(ptr + 0x13_0218)	MISS	HIT	NOT ENOUGH INFO

Problem 5: MIPS Assembly (10 points)

Here is mips assembly for a function with the signature (in C):

```
int f(char* a, char* b);
```

Remember, in a MIPS function call, ra contains the return address, a0 and a1 contain the first and second arguments, respectively, and upon return v0 contains the return value. The s# registers must be preserved across procedure calls. Also remember that standard MIPS has a branch delay slot.

```
00000000 <f>:
   0:   lb      v1,0(a0)
   4:   lb      v0,0(a1)
   8:   addiu   a0,a0,1
   c:   subu    v0,v1,v0
  10:   bnez    v0,20 <f+0x20>
  14:   addiu   a1,a1,1
  18:   bnez    v1,0 <f>
  1c:   nop
  20:   jr      ra
  24:   nop
```

(1) *(7 points)* Explain briefly, in english, what this function does.

(2) *(3 points)* Give one optimization that can be performed on the assembly code.

Computer Architecture Comprehensive Exam (Answers)

Exam Instructions
Answer each of the questions included in the exam. Write all of your answers directly on the examination paper, including any work that you wish to be considered for partial credit. The examination is open-book, and you may make use of the text, handouts, your own course notes, and a calculator.

On equations: Wherever possible, make sure to include the equation, the equation rewritten with the numerical values, and the final solution. Partial credit will be weighted appropriately for each component of the problem, and providing more information improves the likelihood that partial credit can be awarded.

On writing code: Unless otherwise stated, you are free to use any of the assembly instructions listed in the Appendix at the back of the book, including pseudoinstructions. You do not need to optimize your MIPS code unless specifically instructed to do so.

On time: You will have one hour to complete this exam. Budget your time and try to leave some time at the end to go over your work. The point weightings correspond roughly to the time each problem is expected to take.

THE STANFORD UNIVERSITY HONOR CODE

The Honor Code is an undertaking of the students, individually and collectively:

1. that they will not give or receive aid in examinations; that they will not give or receive unpermitted aid in class work, in the preparation of reports, or in any other work that is to be used by the instructor as the basis of grading;
2. that they will do their share and take an active part in seeing to it that others as well as themselves uphold the spirit and letter of the Honor Code.

I acknowledge and accept the Honor Code.

Magic Number: _____

		Score	*Grader*
1. Short Answer	(15)	_____	_____
2. Pipelining	(15)	_____	_____
3. Memory Heirarchy	(15)	_____	_____
4. Cache Math	(5)	_____	_____
5. MIPS Assembly	(10)	_____	_____
TOTAL	(60)	_____	

Problem 1: Short Answer (15 points)

Please provide short, concise answers.

(1) *(2 points)* Your company, Acme Corp., is deciding between two computer systems to deploy its new killer Road Runner Tracking application. Ben Bitdiddle says that his company's system has the better performance because it has the higher clock speed and the higher IPC. Explain why his logic is flawed.

To evaluate performance, you should use execution time for the applications of interest. CPI, IPC, Ghz, FLOPS, IPS, and benchmarks all can imply that processor A is faster than processor B when processor B actually executes the user's application faster.

(2) *(2 points)* Some RISC architectures require the compiler (or assembly programmer) to guarantee that a register not be accessed for a given number of cycles after it is loaded from memory. Give an advantage and a disadvantage of this design choice.

The advantages stem from a simplification of the pipeline logic, which can improve power consumption, area consumption, and maximum speed. The disadvantages are an increase in compiler complexity and a reduced ability to evolve the processor implementation independently of the instruction set (newer generations of chip might have a different pipeline depth and hence a different optimal number of delay cycles).

(3) *(2 points)* Ben Bitdiddle is writing an optimizing compiler for an architecture that supports virtual memory. He notices that his target processor can execute an unaligned 32 bit load more quickly than an 8 bit load. Is it okay for his compiler to emit 32 bit loads and then ignore the extra 24 bits? Why or why not?

No. If a load occurs near the end of a page the extra three bytes may trigger a page fault.

(4) *(3 points)* Briefly describe the data access pattern of an application for which an LRU (least recently used) cache replacement policy performs worse than a random replacement policy.

One pathological case for LRU would be an application that makes multiple sequential passes through a data set that is slightly too large to fit in the cache. The LRU policy will result in a 0% cache hit rate, while under random replacement the hit rate will be high.

(5) *(3 points)* Briefly describe the data access pattern of an application for which a cache with a random replacement policy performs worse than an LRU replacement policy.

A case where LRU would be expected to outperform random replacement is a randomly-accessed tree structure that is too large to fit in the cache. The LRU policy will do a better job of keeping the root of the tree in the cache and those nodes of the tree are accessed more often.

(6) *(3 points)* Briefly give two ways in which loop unrolling can increase performance and one in which it can decrease performance.

Performance can be increased by:

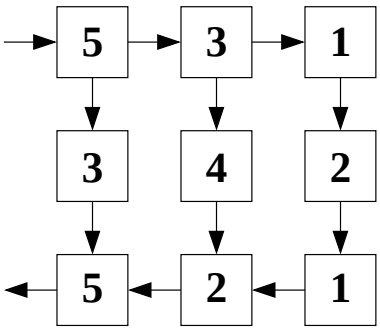
- *Fewer loop conditional evaluations*
- *Fewer branches/jumps*
- *Opportunities to reorder instructions across iterations*
- *Opportunities to merge loads and stores across iterations*

Performance can be decreased by:

- *Increased pressure on the I-cache*
- *Large branch/jump distances may require slower instructions*

Problem 2: Pipelining (15 points)

The National Security Agency has designed a new encryption device called the Conundrum, composed of nine combinational modules connected as shown in the diagram below:

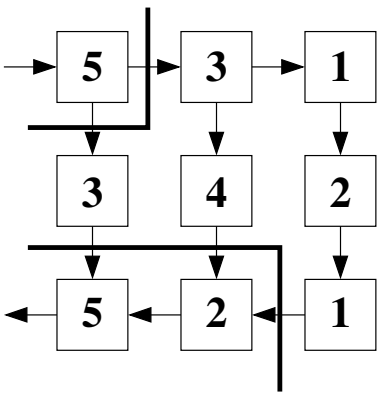


The device takes an integer value X and computes an encrypted version C(X). In the diagram above each combinational component is marked with its propagation delay in microseconds; contamination delays are zero for each component. Assume an ideal (zero-delay) register on both the input and the output of the device.

(1) *(4 points)* What is the latency and throughput of the Conundrum device?

latency (μs) __19__ throughput ($1/\mu s$) __1/19__

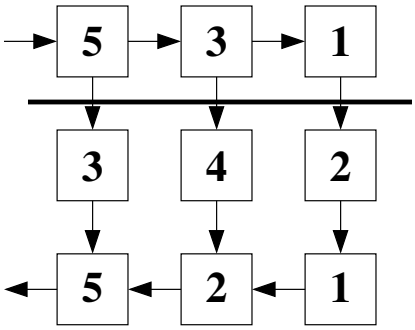
(2) *(9 points)* The NSA needs to produce a version of the Conundrum device that has a throughput larger than 1/15 but wants the implementation with smallest latency (cycle time * pipeline depth) that meets the throughput constraint. Using the diagram below indicate the locations of ideal (zero-delay) registers to create a pipelined implementation that meets these goals. You may use as many registers as necessary. We'll give partial credit for designs with throughput larger than 1/15; full credit for achieving the smallest possible latency. (Extra copies of this diagram can be found on the following page, but only solutions written on the diagram below will be graded).



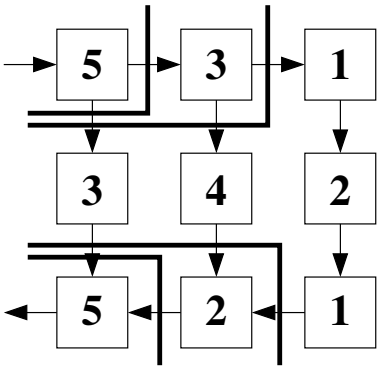
(3) *(2 points)* What is the latency of your pipelined implementation?

latency (μs) __21__ throughput ($1/\mu s$) __1/7__

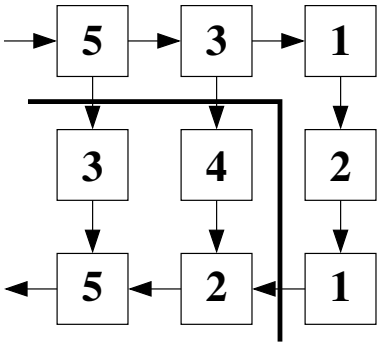
Some valid pipelines with throughput larger than 1/15, but with sub-optimal latency, are given here.



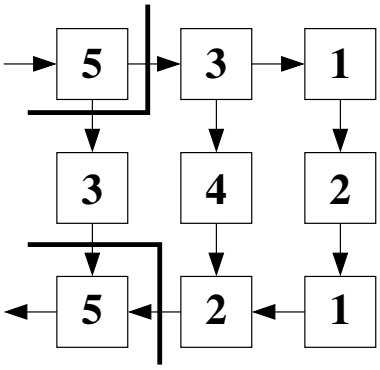
latency: 22
throughput: 1/11



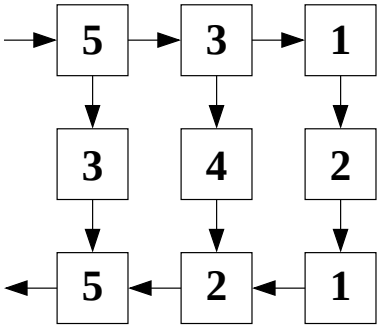
latency: 25
throughput: 1/5
(best possible throughput)

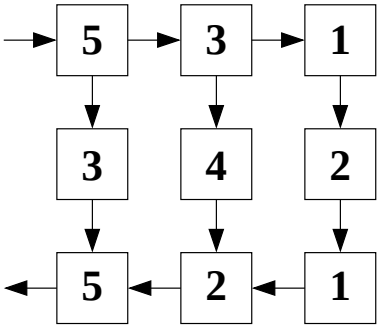


latency: 24
throughput: 1/12



latency: 27
throughput: 1/9





(These diagrams are for scratch work; no solution written here will be graded. Record your solution on the previous page.)

Problem 3: Memory Heirarchy (15 points)

Assume you have a 1 GHz processor with 2-levels of cache and DRAM main memory. The first level cache is split for instructions and data. The system does not use early restart or critical word first, i.e. data blocks must be completely transfered before their results are available. The memory system has the following parameters (Note that here 1KB = 1024 bytes):

	Hit Time	Miss Rate	Block Size
Level-1 cache	1 cycle	6% for data 2% for instruction	32 bytes
Level-2 cache	12 cycles + (1 cycle per 64 bits)	2%	256 bytes
DRAM	70ns + (10ns per 8 bytes)	–	–

The system includes a TLB with a miss rate of 0.5% for data and never incurs a TLB miss for instructions. The TLB miss penalty is 300 cycles and TLB hits take place in parallel with level-1 cache access. All caches in the system are virtually indexed and physically tagged. Assume that the system never swaps memory out to disk.

(1) *(5 points)* What is the average memory access time (AMAT) in clock cycles for instructions?

First you need to calculate the AMAT at every level, taking into account the block size at the highest level and the miss rate and miss penalty at the lower level. Note that at 1 GHZ, 1ns = 1 cycle.

$$
\begin{aligned}
AMAT_{DRAM} &= 70ns + 10ns * (256/8) = 390ns = 390 \; cycles \\
AMAT_{L2} &= hit\ time + miss\ rate * miss\ penalty = (12 + (32/8)) + 0.02 * 390 = 23.8 \; cycles \\
AMAT_{L1} &= hit\ time + miss\ rate * miss\ penalty = (1 + 0.02 * 23.8) = 1.476 \; cycles
\end{aligned}
$$

Thus the overall AMAT is ***1.476 cycles****.*

(2) *(5 points)* What is the AMAT in clock cycles for <u>data</u>? Assume all data accesses are loads.

Only two things change when considering data accesses: the different L1 miss rate and the TLB misses. Note that since the TLB is accessed in parallel with the L1, its hit time is 0 cycles.

$$\begin{aligned} AMAT_{L1} &= hit\ time + miss\ rate * miss\ penalty = (1 + 0.06 * 23.8) = 2.428\ cycles \\ AMAT_{TLB} &= hit\ time + miss\ rate * miss\ penalty = (0 + 0.005 * 300) = 1.5\ cycles \end{aligned}$$

Thus the overall AMAT is $AMAT_{L1} + AMAT_{TLB} =$ ***3.928 cycles***.

(3) *(5 points)* Suppose that we measure the following instruction mix for a program:

Loads: 25%, Stores: 15%, Integer: 30%, Floating-Point: 20%, Branches: 10%

Assume that the processor is using the 5-stage pipeline (base CPI of 1.0). Data hazards cause an average penalty of 0.9 cycles for floating point operations. Integer operations run at maximum throughput. The processor uses the predict-branch-not-taken technique, which turns out to be correct for 80% of the branches. For the remaining branches, there is a 1 cycle stall. What is the average CPI of this program including memory misses (from questions a and b)? Assume that stores have the same AMAT as loads.

CPI = (base CPI) + (CPI due to branch hazards) + (CPI due to FP hazards) + (CPI due to instruction accesses) + (CPI due to data accesses)
We look at each of these in turn:

component	*explanation*	*value*
base	*Given in problem*	*1*
branch	*branch % * misprediction % * misprediction penalty*	$0.1 * 0.2 * 1 = .02$
FP	*FP % * avg penalty*	$0.2 * 0.9 = .18$
instructions	*AMAT for instructions minus 1 to take out the 1 cycle of L1 cache hit that is included in the base CPI*	$1.476 - 1 = 0.476$
data	*(load % + store %) * AMAT for data adjusted as above*	$(0.25 + 0.15) * (3.928 - 1) = 1.171$

This gives us a total CPI of ***2.847***

Problem 4: Cache Math (5 points)

Answers to this question may be in the form of a bare number (decimal or hex), 2^n, or nk (= $n*2^{10} = n*1024$), for example, 65536, 0x10000, 2^{16}, or 64k are all equivalent and acceptable answers.

Consider a 256kb 4-way set associative cache with 256 byte cache lines for a processor that uses 64-bit data words and 48-bit byte addresses. Assume the variable x, of type uint64_t, is stored in memory at location 0x4A85_B413_A518.

(1) *(2 points)* Fill in the bit ranges in the following diagram. Bit ranges should be inclusive. For example, if a field uses bits 0, 1, 2, and 3, label it 3:0.

64-bit word / 8 bits per byte = 8 bytes per word ⇒ 3 bits to select byte in word
256 byte line / 8 bytes per word = 32 words per line ⇒ 5 bits to select word in line
256k cache / 256 bytes per line / 4 ways per set = 256 sets ⇒ 8 bits to select set
The rest of the bits are the tag ⇒ 32 bit tag.

47 : 16	*15 : 8*	*7 : 3*	*2 : 0*
tag	set	word in line	byte in word

(2) *(3 points)* Assume that x is present in the cache, and char* ptr = 0x4A85_B400_0000. Determine if the following accesses will cause a cache miss. For each access circle MISS if it must cause a miss, HIT if it will never cause a miss, or NOT ENOUGH INFO.

*(ptr + 0x11_A538)	MISS	HIT	**(NOT ENOUGH INFO)**
*(ptr + 0x13_A588)	MISS	**(HIT)**	NOT ENOUGH INFO
*(ptr + 0x13_0218)	MISS	HIT	**(NOT ENOUGH INFO)**

**(ptr + 0x11_A538) is in the same set but has a different tag, it may or may not be in another way in that set. *(ptr + 0x13_A588) is in same the line as x (it is in the same set and has the same tag), and since x in present in the cache, it must be also. *(ptr + 0x13_0218) has the same tag, but is in a different set, so it may or may not be in the cache.*

Problem 5: MIPS Assembly (10 points)

Here is mips assembly for a function with the signature (in C):

```
int f(char* a, char* b);
```

Remember, in a MIPS function call, ra contains the return address, a0 and a1 contain the first and second arguments, respectively, and upon return v0 contains the return value. The s# registers must be preserved across procedure calls. Also remember that standard MIPS has a branch delay slot.

```
00000000 <f>:
   0:   lb      v1,0(a0)
   4:   lb      v0,0(a1)
   8:   addiu   a0,a0,1
   c:   subu    v0,v1,v0
  10:   bnez    v0,20 <f+0x20>
  14:   addiu   a1,a1,1
  18:   bnez    v1,0 <f>
  1c:   nop
  20:   jr      ra
  24:   nop
```

(1) *(7 points)* Explain briefly, in english, what this function does.

The function compares the two strings a and b. It returns an integer less than, equal to, or greater than zero if a is found, respectively, to be less than, to match, or be greater than b. In other words, f() is strcmp(). Note that the comparison of characters is done using 8-bit signed arithmetic.

(2) *(3 points)* Give one optimization that can be performed on the assembly code.

The instruction at location 0x8 (addiu a0,a0,1) can be moved into the branch delay slot at location 0x1c. Note that the instruction at 0x14 is already in a branch delay slot, so moving it to the other branch delay slot does not improve the code.

Stanford University Computer Science Department
2007 Comprehensive Exam in Databases

- The exam is *open book and notes*, but not open computer.
- There are six problems on the exam, with a varying number of points for each problem and subproblem for a total of 60 points (i.e., one point per minute). It is suggested that you look through the entire exam before getting started, in order to plan your strategy.
- Please write your solutions in the spaces provided on the exam. Make sure your solutions are neat and clearly marked.
- *Simplicity and clarity of solutions will count.* You may get as few as 0 points for a problem if your solution is far more complicated than necessary, or if we cannot understand your solution.

MAGIC NUMBER: _____

Problem	1	2	3	4	5	6	TOTAL
Max. points	10	8	12	18	3	9	60
Points							

1. *Data Modeling* **(10 points)**

   Design an Entity/Relationship diagram for a database that holds the following information:

   i. There are teams. Each team has a unique team name, a city in which it plays, and a mascot.

   ii. There are players. Each player has a name, a number, and a position. Each player plays for a unique team.

   iii. Teams assign unique numbers to their players, but two teams may use the same numbers.

   iv. Each team has at most one team captain, who is a player.

   v. The day on which two teams play, and which team is the home team if they do play. You may assume no pair of teams plays more than once.

   Your diagram should be as succinct as possible, but must capture all the above information. If you find the specifications ambiguous, appeal to the way the world actually works, e.g., it is entirely possible that two players have the same name.

2. *Dependency Theory* **(8 points)**

   The following functional dependencies hold in relation $R(A, B, C, D, E, F)$:

   $AB \rightarrow C$
   $DE \rightarrow F$
   $F \rightarrow B$
   $C \rightarrow E$

   Find all the keys for R. Explain your reasoning for partial credit.

3. *Relational Algebra* **(12 points)**

   Suppose we have a relation $T(P, C, I)$ representing a tree. That is, nodes are represented by integers, and the meaning of a tuple (p, c, i) in T is that node c is the ith child of node p, from the left. We want you to write certain queries in **relational algebra**, using only the basic operations: select, project, natural and theta join, product, renaming, union, intersection, and difference. You may define temporary relations if you like; i.e., you may break an expression into a sequence of steps. To receive full credit, your expressions should be (close to) as simple as possible.

   (a) (3 pts.) Write a query to produce the set of distinct pairs of nodes that are siblings (i.e., they have the same parent).

(b) (3 pts.) If you can use arithmetic in selection conditions, e.g., $x = y + 1$, then it is relatively easy to write a query that produces the set of pairs (a, b) such that b is the right-sibling of a; that is, a and b have the same parent, and b is immediately to the right of a among the children of their parent. Write this query.

(c) (6 pts.) But you do not need arithmetic to write the query of part (b). Write the same query, using only conditions of the form $x\theta y$, where θ is a single comparison operator (e.g., = or $<$) and x and y are attributes or constants. Both selection and theta-join are limited in this way.

4. *SQL* **(18 points, 3 each part)**

Two queries are *equivalent on a database instance* if they produce the same result relation when evaluated over that instance.

Consider a SQL database with two tables, $R(A, B)$ and $S(C)$. Assume NULL values are not permitted in any attribute; make no other assumptions about the data.

Below are pairs of queries. For each pair, select one of the following three options:

- **EQ** – The two queries are equivalent on every possible instance of R and S.
- **NEQ** – There are no instances of R and S on which the two queries are equivalent.
- **SEQ** – The two queries are equivalent for some instances of R and S, but not all. When choosing this option, state conditions over the data in R and S that ensure the two queries are equivalent. Try to make the conditions as general as possible, meaning whenever the conditions are violated the queries may not be equivalent. (E.g., "The tables are empty" is not a good choice.) The conditions should be stated succinctly and should use SQL constraint terminology (e.g., "A is a key") when applicable.

(a)
```
Query1: Select A From R
Query2: Select A From R, S
```

EQ, **NEQ**, or **SEQ** with condition:

(b)
```
Query1: Select A From R
Query2: Select A From R, S Where B = C
```

EQ, **NEQ**, or **SEQ** with condition:

(continued on next page)

(c)
```
Query1: Select Distinct A From R
Query2: Select A From R Group By A
```

EQ, **NEQ**, or **SEQ** with condition:

(d)
```
Query1: Select count(*) From R
Query2: Select count(Distinct A) From R
```

EQ, **NEQ**, or **SEQ** with condition:

(e)
```
Query1: Select max(A) From R
Query2: Select A From R Where A >= all (select A from R)
```

EQ, **NEQ**, or **SEQ** with condition:

(f)
```
Query1: Select Distinct A From R Group By A Having max(B) > 10
Query2: Select Distinct A from R Where B > 10
```

EQ, **NEQ**, or **SEQ** with condition:

5. *Views* **(3 points)**

Is it possible to create an index on a SQL view? Answer *yes* or *no*, with a brief explanation.

6. *Transactions* **(9 points, 3 each part)**

Consider two tables $R(A, B)$ and $S(C)$ as in Problem 4. Below are pairs of transactions. For each pair, decide whether it is possible for *nonserializable* behavior to be exhibited when executing the transactions together, while respecting their specified isolation levels. Assume each transaction completes successfully.

(a)
```
Transaction 1: {
  Set Transaction Isolation Level Read Committed;
  Select count(*) From R;
  Select count(*) From S;
  Commit;
  }
Transaction 2: {
  Set Transaction Isolation Level Serializable;
  Insert Into R Values (1,2);
  Insert Into S Values (3);
  Commit;
  }
```

Nonserializable behavior possible (*yes* or *no*)? [ ]

(b)
```
Transaction 1: {
  Set Transaction Isolation Level Read Committed;
  Select count(*) From R;
  Select count(*) From S;
  Commit;
  }
Transaction 2: {
  Set Transaction Isolation Level Serializable;
  Insert Into R Values (1,2);
  Insert Into R Values (3,4);
  Commit;
  }
```

Nonserializable behavior possible (*yes* or *no*)? [ ] ** *one more on next page*

(c)
```
Transaction 1: {
  Set Transaction Isolation Level Repeatable Read;
  Select count(*) From R;
  Select count(*) From S;
  Select count(*) From R;
  Commit;
}
Transaction 2: {
  Set Transaction Isolation Level Serializable;
  Insert Into R Values (1,2);
  Commit;
  }
```

Nonserializable behavior possible (*yes* or *no*)? [ ]

Stanford University Computer Science Department
2007 Comprehensive Exam in Databases
Sample Solution

1. *Data Modeling* **(10 points)**

   Design an Entity/Relationship diagram for a database that holds the following information:

   i. There are teams. Each team has a unique team name, a city in which it plays, and a mascot.

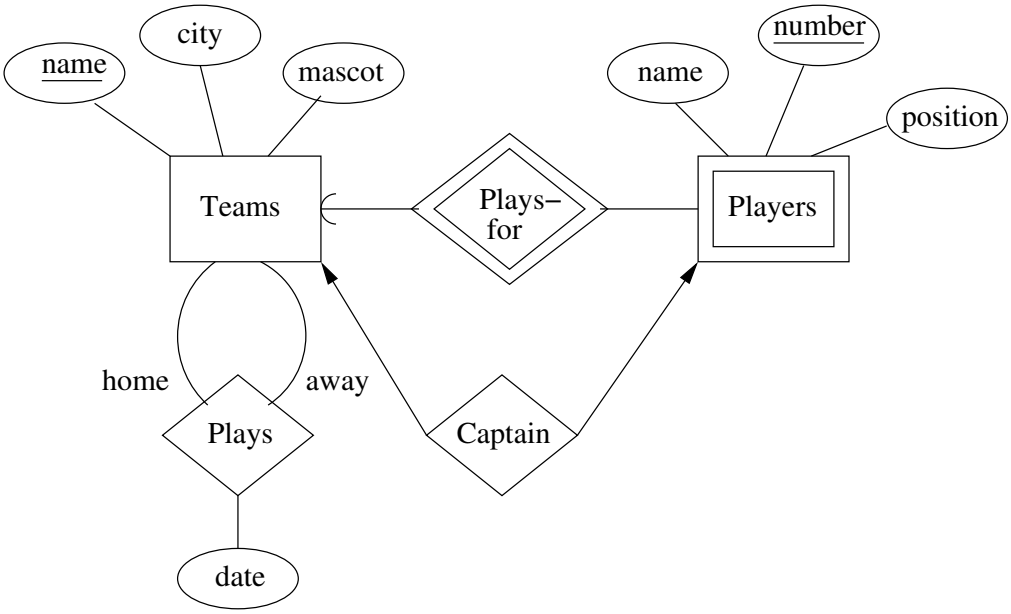
   ii. There are players. Each player has a name, a number, and a position. Each player plays for a unique team.

   iii. Teams assign unique numbers to their players, but two teams may use the same numbers.

   iv. Each team has at most one team captain, who is a player.

   v. The day on which two teams play, and which team is the home team if they do play. You may assume no pair of teams plays more than once.

   Your diagram should be as succinct as possible, but must capture all the above information. If you find the specifications ambiguous, appeal to the way the world actually works, e.g., it is entirely possible that two players have the same name.



2. *Dependency Theory* **(8 points)**

   The following functional dependencies hold in relation $R(A, B, C, D, E, F)$:

   $AB \rightarrow C$
   $DE \rightarrow F$
   $F \rightarrow B$
   $C \rightarrow E$

   Find all the keys for R. Explain your reasoning for partial credit.

   Since A and D do not appear on the right of any FD, they must be in every key. They are not a key by themselves, but if you add any other attribute and take the closure, you get all attributes. Thus, there are four keys: ABD, ACD, ADE, and ADF.

3. *Relational Algebra* **(12 points)**

   Suppose we have a relation $T(P, C, I)$ representing a tree. That is, nodes are represented by integers, and the meaning of a tuple (p, c, i) in T is that node c is the ith child of node p, from the left. We want you to write certain queries in **relational algebra**, using only the basic operations: select, project, natural and theta join, product, renaming, union, intersection, and difference. You may define temporary relations if you like; i.e., you may break an expression into a sequence of steps. To receive full credit, your expressions should be (close to) as simple as possible.

   (a) (3 pts.) Write a query to produce the set of distinct pairs of nodes that are siblings (i.e., they have the same parent).

   $$\pi_{C1,C2}\Big(\sigma_{I1 \neq I2}\big(\rho_{T1(P,C1,I1)}(T) \bowtie \rho_{T2(P,C2,I2)}(T)\big)\Big)$$

   (b) (3 pts.) If you can use arithmetic in selection conditions, e.g., $x = y + 1$, then it is relatively easy to write a query that produces the set of pairs (a, b) such that b is the right-sibling of a; that is, a and b have the same parent, and b is immediately to the right of a among the children of their parent. Write this query.

   $$\pi_{C1,C2}\Big(\sigma_{I2 = I1+1}\big(\rho_{T1(P,C1,I1)}(T) \bowtie \rho_{T2(P,C2,I2)}(T)\big)\Big)$$

(c) (6 pts.) But you do not need arithmetic to write the query of part (b). Write the same query, using only conditions of the form $x\theta y$, where θ is a single comparison operator (e.g., = or $<$) and x and y are attributes or constants. Both selection and theta-join are limited in this way.

$$R1(C1, C2) := \pi_{C1,C2}\Big(\sigma_{I2>I1}(\rho_{T1(P,C1,I1)}(T) \bowtie \rho_{T2(P,C2,I2)}(T))\Big)$$
$$R2(C1, C3) := \pi_{C1,C3}\Big(\sigma_{I2>I1\ AND\ I3>I2}(\rho_{T1(P,C1,I1)}(T) \bowtie \rho_{T2(P,C2,I2)}(T) \bowtie \rho_{T3(P,C3,I3)}(T))\Big)$$
$$ANSWER := R1 - R2$$

4. *SQL* **(18 points, 3 each part)**

Two queries are *equivalent on a database instance* if they produce the same result relation when evaluated over that instance.

Consider a SQL database with two tables, $R(A, B)$ and $S(C)$. Assume NULL values are not permitted in any attribute; make no other assumptions about the data.

Below are pairs of queries. For each pair, select one of the following three options:

- **EQ** – The two queries are equivalent on every possible instance of R and S.
- **NEQ** – There are no instances of R and S on which the two queries are equivalent.
- **SEQ** – The two queries are equivalent for some instances of R and S, but not all. When choosing this option, state conditions over the data in R and S that ensure the two queries are equivalent. Try to make the conditions as general as possible, meaning whenever the conditions are violated the queries may not be equivalent. (E.g., "The tables are empty" is not a good choice.) The conditions should be stated succinctly and should use SQL constraint terminology (e.g., "A is a key") when applicable.

(a)
```
Query1: Select A From R
Query2: Select A From R, S
```

EQ, **NEQ**, or **SEQ** with condition:

SEQ
Condition: S contains exactly one tuple

(b)
```
Query1: Select A From R
Query2: Select A From R, S Where B = C
```

EQ, **NEQ**, or **SEQ** with condition:

SEQ
Condition: *Referential integrity* from $R.B$ (foreign key) to $S.C$ (primary key)

(c)
```
Query1: Select Distinct A From R
Query2: Select A From R Group By A
```

EQ, **NEQ**, or **SEQ** with condition:

EQ

(d)
```
Query1: Select count(*) From R
Query2: Select count(Distinct A) From R
```

EQ, **NEQ**, or **SEQ** with condition:

SEQ
Condition: A is a key

(e)
```
Query1: Select max(A) From R
Query2: Select A From R Where A >= all (select A from R)
```

EQ, **NEQ**, or **SEQ** with condition:

SEQ
Condition: The maximum A value has no duplicates

(f)
```
Query1: Select Distinct A From R Group By A Having max(B) > 10
Query2: Select Distinct A from R Where B > 10
```

EQ, **NEQ**, or **SEQ** with condition:

EQ

5. *Views* **(3 points)**

Is it possible to create an index on a SQL view? Answer *yes* or *no*, with a brief explanation.

No. SQL views are not indexable tables; they are used only for rewriting queries.
(Also accepted for full credit: If a view is *materialized* it can be indexed, but if it is *virtual* it cannot be.)

6. *Transactions* **(9 points, 3 each part)**

Consider two tables $R(A, B)$ and $S(C)$ as in Problem 4. Below are pairs of transactions. For each pair, decide whether it is possible for *nonserializable* behavior to be exhibited when executing the transactions together, while respecting their specified isolation levels. Assume each transaction completes successfully.

(a)
```
Transaction 1: {
  Set Transaction Isolation Level Read Committed;
  Select count(*) From R;
  Select count(*) From S;
  Commit;
  }
Transaction 2: {
  Set Transaction Isolation Level Serializable;
  Insert Into R Values (1,2);
  Insert Into S Values (3);
  Commit;
  }
```
Nonserializable behavior possible (*yes* or *no*)? **Yes**

(b)
```
Transaction 1: {
  Set Transaction Isolation Level Read Committed;
  Select count(*) From R;
  Select count(*) From S;
  Commit;
  }
Transaction 2: {
  Set Transaction Isolation Level Serializable;
  Insert Into R Values (1,2);
  Insert Into R Values (3,4);
  Commit;
  }
```
Nonserializable behavior possible (*yes* or *no*)? **No**

(c)
```
Transaction 1: {
  Set Transaction Isolation Level Repeatable Read;
  Select count(*) From R;
  Select count(*) From S;
  Select count(*) From R;
  Commit;
}
Transaction 2: {
  Set Transaction Isolation Level Serializable;
  Insert Into R Values (1,2);
  Commit;
  }
```
Nonserializable behavior possible (*yes* or *no*)? **Yes**

Computer Graphics Comprehensive Exam

2007-2008

NAME:

Note: This is exam is *closed-book*.

The exam consists of 5 questions. Each question is worth 20 points. Please answer all the questions in the space provided, overflowing on to the back of the page if necessary.

This exam has been designed to take 1 hr.

1. [20 points] General.

1A [5 points]. What is the relationship between the RGB and CMY color spaces? How are colors in one space computed from colors in the other space?

1B [5 points]. What is a z-buffer and how is it used in computer graphics?

1C [5 points]. What is quantization? And why can it be used for compression?

1D [5 points]. Aliasing is an undesirable artifact of generating images using computers. What causes aliasing?

[20 points] Shading Polygon Meshes

Polygon meshes are the most common representation of geometric shapes in computer graphics. Many computer graphics algorithms operate on polygon meshes. For this question, assume that the mesh is made of polygons with any number of sides.

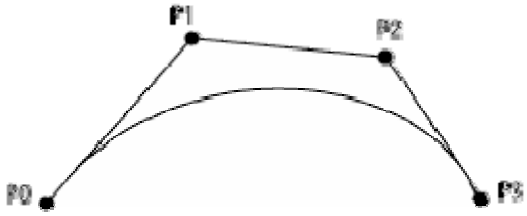
2A [5 points]. What is the difference between Gouraud shading and Phong shading?

All shading algorithms require a surface normal. In the case of Phong and Gouraud shading, a surface normal is required for each vertex in the mesh. Assume that you are given a polygon mesh without normals. That is, with just a position for each vertex.

2B [15 points]. Describe a method for computing the normal at each vertex. Your explanation should explain the math of how to compute the normal from the vertex positions (7.5 points) and how to define a polygon mesh data structure that makes this computation efficient (O(n) where n is the number of vertices in the mesh).

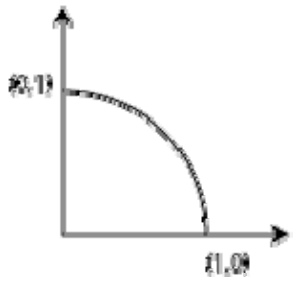
3. [20 points] Bezier Curves.

One the most important primitives in computer graphics is the cubic Bezier Curve. The cubic Bezier curve is constructed from a control polygon consisting of 4 points as shown below.



The curve is given by the parametric equation P(t) where t lies between 0 and 1. This diagram illustrates two key properties of Bezier curves. First, the curve passes through the end points P0 and P3. That is, P(0)=P0 and P(1) = P3. Second, the curve is tangent to the control polygon at the end points. Precisely, T(0)=3(P1-P0) and T(1)=3(P3-P2). P(t) can be evaluated using a recursive algorithm. Recall the classic result that P(1/2) = 1/8 P0 + 3/8 P1 + 3/8 P2 + 1/8 P3.

One reason Bezier curves are so popular is that they can be used to accurately approximate other curves. Suppose you want to approximate the circular arc shown below. Where would you position the control points P0, P1, P2, P3 to approximate this arc with a bezier curve whose P(1/2) is the arc's midpoint (cos 45, sin 45)? Provide the coordinates of the positions of these points.



4. [20 points] Shading

There are two important shading models used to model the appearance of real materials, the diffuse and specular (sometimes called glossy) models. In this problem, just consider point light sources.

4A [10 points]. What is the equation for diffuse reflection? That is, what is the reflected color given the direction to the eye **E**, the surface normal **N**, and the direction of the point light source **L**. Include other parameters that might be useful. Explain briefly the physics underlying diffuse reflection. Finally, give an example of a material whose appearance can be modeled using diffuse reflection.

4B [10 points]. Give an equation for specular or glossy reflection. Explain briefly the physics underlying this type of reflection. Give an example of a material that can be modeled using specular reflection.

5 [20 points] Matrices and the Matrix Stack

5A [5 points]. Transformations in computer graphics are represented with 4x4 matrices even though the world is three-dimensional. Why are 4x4 matrices used instead of 3x3 matrices?

In a typical graphics system, a current transformation matrix T is part of the graphics state. Each time a triangle or line is drawn, this matrix transforms the vertices. Assume that the vertex position p is given by a column vector. With this assumption, the transformation involves a matrix-vector product, T * p.

5B [5 points]. In a graphics program, commands exist to modify the current transformation. For example, translate(x,y,z) and rotatex(30). What happens in the graphics system when a transformation command is given? Be precise.

5C [5 points]. The order of the transformations sometimes matters. That is, a program that first executes translate(x,y,z) and then executes rotatex(30) may lead to a result different than one that first executes rotatex(30) and then translate(x,y,z). Why does the order matter? If order does not always matter, can you give an example when it does not matter?"

5D [5 points]. Commands also exist to push and pop the current transformation onto a stack. Why is it useful to be able push and pop the current transformation?

Computer Graphics Comprehensive Exam

2007-2008

NAME:

Note: This is exam is *closed-book.*

The exam consists of 5 questions. Each question is worth 20 points. Please answer all the questions in the space provided, overflowing on to the back of the page if necessary.

This exam has been designed to take 1 hr.

1. [20 points] General.

1A [5 points]. What is the relationship between the RGB and CMY color spaces? How are colors in one space computed from colors in the other space?

RGB is an additive color space, CMY is a subtractive coor space.

R = 1-C
G = 1-M
B = 1-Y

(Need these formulas to get full credit)

1B [5 points]. What is a z-buffer and how is it used in computer graphics?

The z-buffer is an image buffer that holds the depth of the closest surface visible at each pixel. The z-buffer is used in hidden surface elimination. As each surface is rasterized, the depth of that fragment is compared with the depth in the z-buffer. If the fragment is closer, it is drawn and the z-buffer is updated. If the fragment is further, it is discarded.

1C [5 points]. What is quantization? And why can it be used for compression?

Quantization is the process of mapping a continuous (higher-precision) value to a discrete (lower-precision) value. A set of values in the input range are associated with a single value in the output range. Since there are fewer values to store, fewer bits are needed to represent the output set of values.

1D [5 points]. Aliasing is an undesirable artifact of generating images using computers. What causes aliasing?

Aliasing is caused by undersampling a signal. Frequencies that are greater than ½ the sampling rate will appear as aliases. Aliases are frequency components that appear at different frequencies.

(Sampling need not generate aliasing; merely creating an image may not cause aliasing.)

[20 points] Shading Polygon Meshes

Polygon meshes are the most common representation of geometric shapes in computer graphics. Many computer graphics algorithms operate on polygon meshes. For this question, assume that the mesh is made of polygons with any number of sides.

2A [5 points]. What is the difference between Gouraud shading and Phong shading?
In Gouraud shading, shading is performed at each vertex to generate a color, and these colors are interpolated over the polygon. In Phong shading, a normal is computed at each vertex, these normals are interpolated over the polygon, and shading is performed at each pixel.

All shading algorithms require a surface normal. In the case of Phong and Gouraud shading, a surface normal is required for each vertex in the mesh. Assume that you are given a polygon mesh without normals. That is, with just a position for each vertex.

2B [15 points]. Describe a method for computing the normal at each vertex. Your explanation should explain the math of how to compute the normal from the vertex positions (7.5 points) and how to define a polygon mesh data structure that makes this computation efficient (O(n) where n is the number of vertices in the mesh).

Three vertices define a triangle and the normal to the triangle can be computed using the vector cross-product. If the vertices are p0, p1, p2, then n = (p1-p0) x (p2-p0).

The normal to a polygon is often computed by averaging the normals from each set of three adjacent vertices. This requires a data structure to store the list of vertices associated with each polygon.

The normal at a vertex is then computed by averaging the normals of the adjacent polygons. This requires a data structure to store the list of polygons adjacent to a vertex.

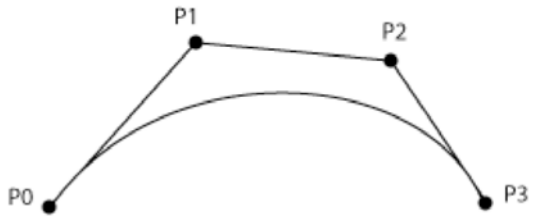
Here is one possible set of data structures:

```
struct { float p[3], n[3]; List *polygons; } Vertex;
struct { List *vertices; } Polygon;
```

There are lots of variations that are acceptable and any reasonable data structure was given full credit.

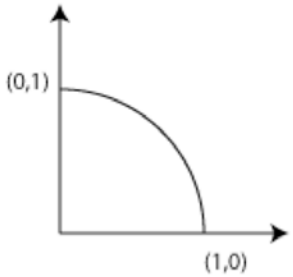
3. [20 points] Bezier Curves.

One the most important primitives in computer graphics is the cubic Bezier Curve. The cubic Bezier curve is constructed from a control polygon consisting of 4 points as shown below.



The curve is given by the parametric equation P(t) where t lies between 0 and 1. This diagram illustrates two key properties of Bezier curves. First, the curve passes through the end points P0 and P3. That is, P(0)=P0 and P(1) = P3. Second, the curve is tangent to the control polygon at the end points. Precisely, T(0)=3(P1-P0) and T(1)=3(P3-P2). P(t) can be evaluated using a recursive algorithm. Recall the classic result that P(1/2) = 1/8 P0 + 3/8 P1 + 3/8 P2 + 1/8 P3.

One reason Bezier curves are so popular is that they can be used to accurately approximate other curves. Suppose you want to approximate the circular arc shown below. Where would you position the control points P0, P1, P2, P3 to approximate this arc with a bezier curve whose P(1/2) is the arc's midpoint (cos 45, sin 45)? Provide the coordinates of the positions of these points.



v = 4/3 (sqrt(2) – 1)

P0 = (1,0)
P1 = (1,v)
P2 = (v,1)
P3 = (0,1)

4. [20 points] Shading

There are two important shading models used to model the appearance of real materials, the diffuse and specular (sometimes called glossy) models. In this problem, just consider point light sources.

4A [10 points]. What is the equation for diffuse reflection? That is, what is the reflected color given the direction to the eye **E**, the surface normal **N**, and the direction of the point light source **L**. Include other parameters that might be useful. Explain briefly the physics underlying diffuse reflection. Finally, give an example of a material whose appearance can be modeled using diffuse reflection.

kd Cl max(L . N, 0), here kd is the diffuse reflection coefficient and Cl is the color of the light.

In diffuse reflection, light is scattered equally in all direction. The factor N . L gives the amount of light falling on the surface.

Chalk is a good example of a diffuse material.

4B [10 points]. Give an equation for specular or glossy reflection. Explain briefly the physics underlying this type of reflection. Give an example of a material that can be modeled using specular reflection.

ks Cl (N . H)^s, here ks is the specular reflection coefficient and H is the vector halfway between L and E (H = (L+E)/len(L + E).

In specular reflection, the surface behaves as a mirror. In the above formula, (N.H)^s gives the number of microfacets (small mirrors) oriented to reflect light.

Plastic and metals are good examples of glossy materials.

Full credit was also given for using ideal specular reflection from a true mirror, and the Phong Reflection Model.

5 [20 points] Matrices and the Matrix Stack

5A [5 points]. Transformations in computer graphics are represented with 4x4 matrices even though the world is three-dimensional. Why are 4x4 matrices used instead of 3x3 matrices?

Adding an extra 4^{th} coordinate allows the use of homogenous coordinates, which permit the representation of translations and perspective transformations as matrices.

In a typical graphics system, a current transformation matrix T is part of the graphics state. Each time a triangle or line is drawn, this matrix transforms the vertices. Assume that the vertex position p is given by a column vector. With this assumption, the transformation involves a matrix-vector product, T * p.

5B [5 points]. In a graphics program, commands exist to modify the current transformation. For example, translate(x,y,z) and rotatex(30). What happens in the graphics system when a transformation command is given? Be precise.

Suppose the current transformation matrix is T, and that Tr represents the translation and R a rotation. Then specifying T causes T = T * Tr, and specifying R causes T = T * R. The matrix is post-multiplied onto the CTM because the CTM premultiplies the position vector.

5C [5 points]. The order of the transformations sometimes matters. That is, a program that first executes translate(x,y,z) and then executes rotatex(30) may lead to a result different than one that first executes rotatex(30) and then translate(x,y,z). Why does the order matter? If order does not always matter, can you give an example when it does not matter?"

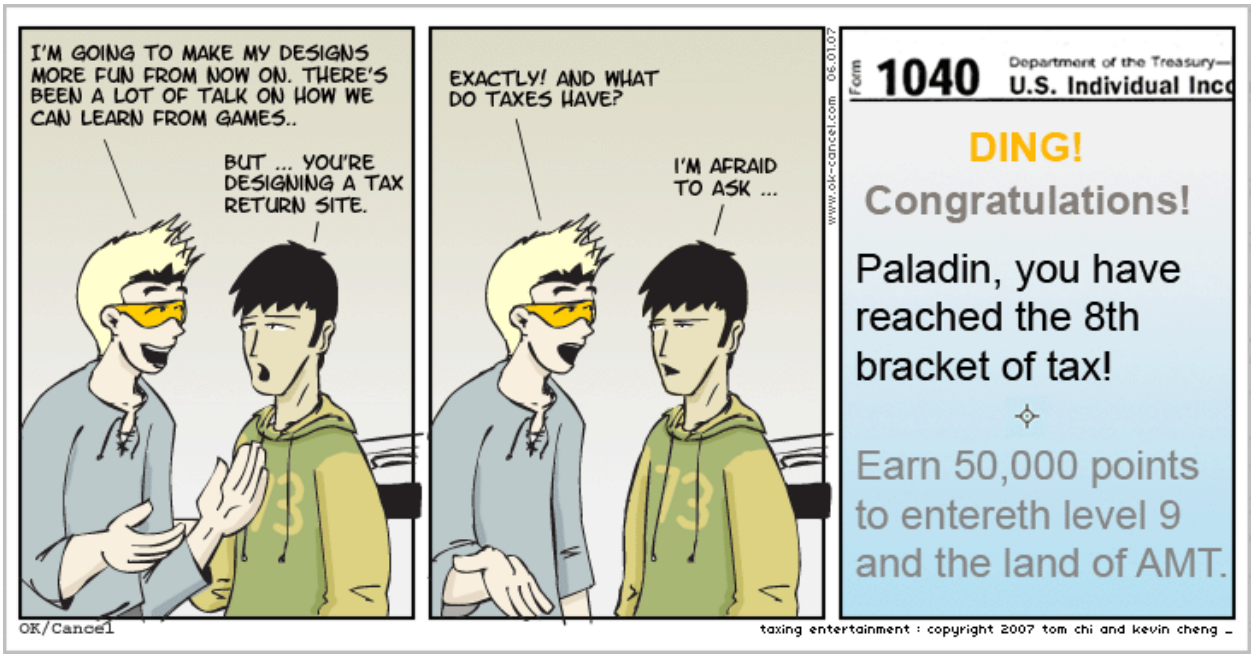
Order matters because in general matrices do not commute; alternatively many transformations do not commute. However, some transformations do commute. For example, two translates will commute, so the order does not matter.

5D [5 points]. Commands also exist to push and pop the current transformation onto a stack. Why is it useful to be able push and pop the current transformation?

Many applications use hierarchical models such as a skeleton. The hierarchies are drawn recursively. When a node in the hierarchy is drawn, the current transformation is pushed, the node drawn, and then the current transformation is popped.

Comprehensive Exam in Human-Computer Interaction

Scott Klemmer, 30 October 2007



1	/ 25
2	/ 16
3	/ 20
4	/ 10
5	/ 10
Total	/ 81

This is a closed book, closed notes exam.

MAGIC NUMBER:

I. A friend is designing the interface to a new operating system, cheeryOS. She has several questions for you. Answer each part in 2-3 sentences. *(25 points total)*

1) First, she asks, "Should I use a graphical user interface or a command line?"
    a. Please begin by using ideas from the Hinckley reading on input to argue for one of the two based purely on error-free expert performance. (*4 points*)

    b. "Of course, expert performance isn't the only consideration," you continue. Explain the two "gulfs" described in the Hutchins *et al.* reading, and advocate for the interaction style that minimizes them. (*4 points*)

    c. What are the two types of distance that contribute to those gulfs? Pick one of those two types, and describe a means of reducing that distance. (*4 points*)

    d. Finally, draw on a concept from another reading that suggests the graphical user interface. Explain both a benefit and a caveat to using this concept. (*4 points*)

2) "Thanks for explaining the difference," your friend says. "I've been thinking about other options as well. What do you think about agents?" Present two ideas from the readings that argue for or against the use of agents in cheeryOS. (*4 points*)

3) "Ah, I see. What if we were to do something more radical, beyond the screen. What if I were to use a tabletop interface, where the input was provided through physical icons, rather than a mouse. From a user experience perspective, how would this be different than a graphical interface?" Draw on the Dourish, Norman, and Hutchins readings. (*4 points*)

"Thank you so much! I think now users can milk cheeryOS for all it's worth…"
(1 point free for enduring the bad pun)

II. User Testing *(16 Points total)*

To make use of your newly acquired expertise in HCI, you volunteered as a peer reviewer for an academic conference, and you were invited to review a paper submission entitled "PARROT: A note-taking support system" Read the following excerpt from the paper's "Evaluation" section.

> To evaluate my system, I invited undergraduate CS students enrolled in my course to participate in an experiment for extra credit. I explained to them that they would be comparing an old system for note-taking (Microsoft Word) to a new data entry system that I had been building for three years. Five participants were asked to enter one page of data, first using Microsoft Word, and then using my PARROT system. I then asked participants to rate both systems on an integer scale from 1 (bad) to 4 (phenomenally awesome). Word received an average rating of 3, while PARROT received an average rating of 3.5. Several subjects commented that PARROT was "easy to use" and "intuitive."

Point out four problems with the experimental methodology, and suggest a way to fix each problem. *(4 points each)*

III. Prototyping *(20 points total)*

You are helping the designer of a home climate control system. The system should enable the user to do things like program the temperature for different times of day and for different days, such as weekends and weekdays, summer versus winter, and so on. Here are three alternate versions that are potential designs.

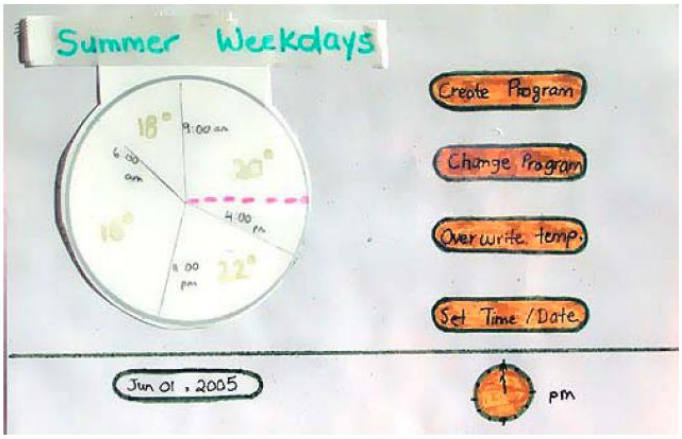


Figure 1. The "Circular" paper prototype

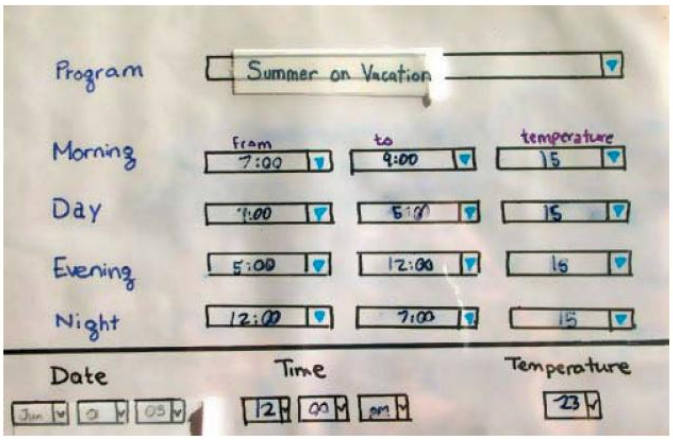


Figure 2. The "Tabular" paper prototype

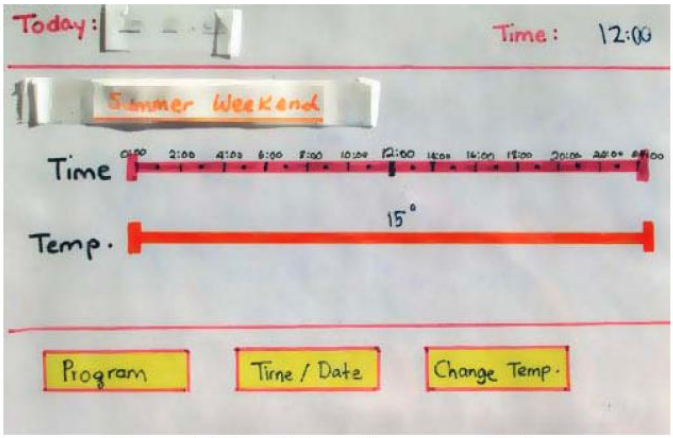


Figure 3. The "Linear" paper prototype

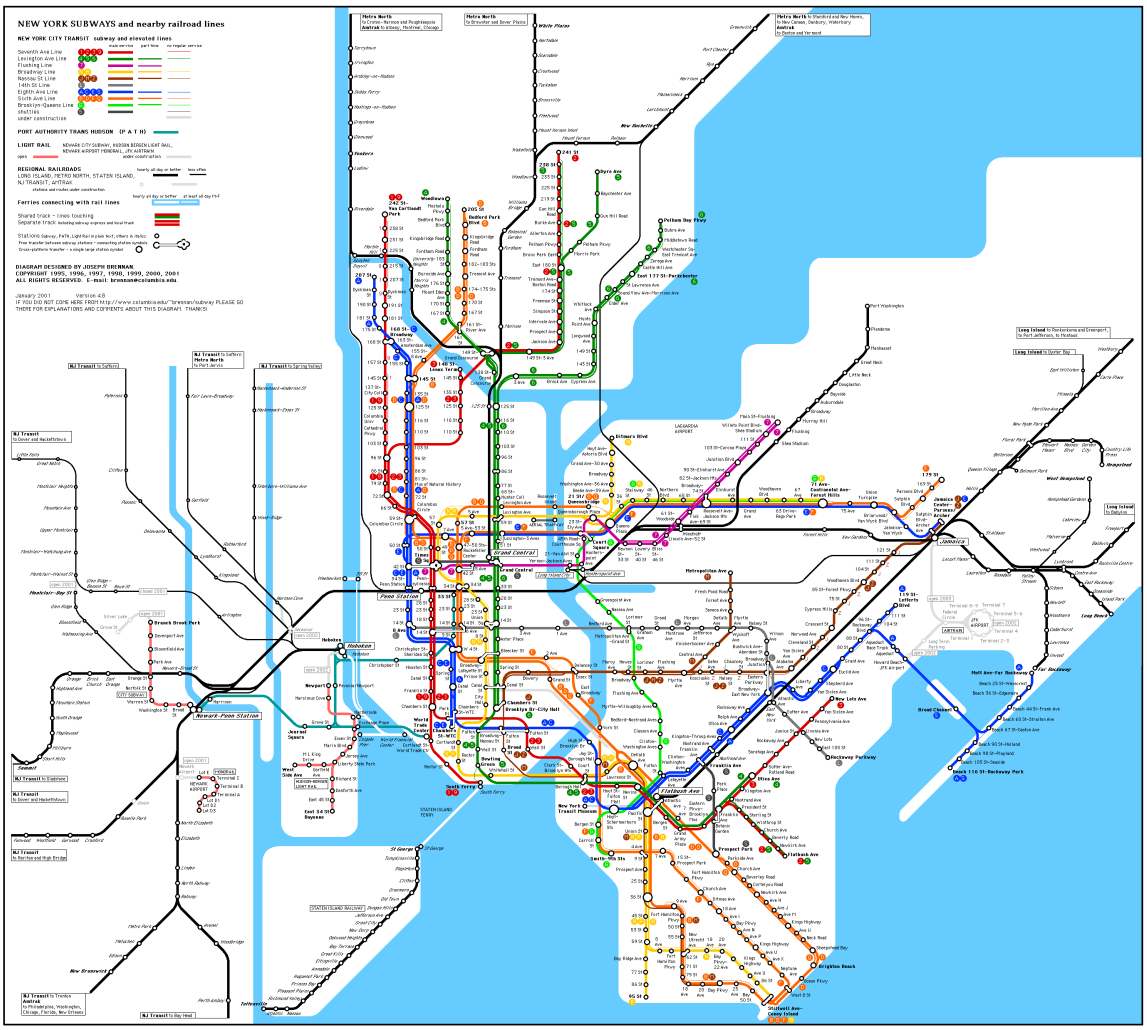
a. In one sentence, use Houde and Hill's model to describe what these prototypes accomplish. *(5 points)*

b. Use the ideas of conceptual models and affordances to describe the relative merits of the different designs. (*5 points*)

c. Ask and answer four standard task analysis questions for a home climate system. (*10 points*)

IV. Interaction Design *(10 points)*

Here is a map of the New York subway system. From the perspective of the readings, write a paragraph that explains what the important differences are between this map and a traditional cartographic map, why those design decisions are made, and when tasks this map is and is not preferable.



V. Evaluation (10 Points)

(a) Circle the **one** answer that **best** completes the sentence *(5 points)*

A **within-subjects** experimental design:

A. means that each participant uses all of the systems being compared.
B. specifies what occurs between the time you test two different participants.
C. means that each participant uses only one of the systems being compared.
D. reduces the variability in the results.
E. is best for testing low-level interaction techniques.

(c) Norman makes the distinction between two types of errors. What are these two types, and what is their distinguishing characteristic. (*5 points*)

2007 Comprehensive Examination
Logic

Honor code and all that stuff.

1. Logical Entailment. (20 points) Let Γ and Δ be sets of closed sentences in first-order logic, and let φ and ψ be individual closed sentences in first-order logic. State whether each of the following statements is true or false. No explanation is necessary.

(a) If $\Gamma \cap \Delta \models \varphi$, then $\Gamma \models \varphi$ and $\Delta \models \varphi$.

(b) If $\Gamma \cup \Delta \models \varphi$, then $\Gamma \models \varphi$ or $\Delta \models \varphi$.

(c) $\Delta \models (\varphi \Rightarrow \psi)$ if and only if $\Delta \cup \{\varphi\} \models \psi$.

(d) $\Delta \models \varphi$ or $\Delta \models \psi$ if and only if $\Delta \models (\varphi \vee \psi)$.

(e) If $\Delta \models \varphi$ and $\Delta \models \neg\psi$, then $\Delta \not\models (\varphi \Rightarrow \psi)$.

(f) If $\Delta \models \varphi$ and $\Delta \models \psi$, then $\Delta \models (\varphi \Rightarrow \psi)$.

(g) If $\Delta \models p(\tau)$ for some ground term τ, then $\Delta \not\models \forall x.\neg p(x)$.

(h) If $\Delta \models p(\tau)$ for every ground term τ, then $\Delta \models \forall x.p(x)$.

(i) If $\Delta \models \forall x.(p(x) \Rightarrow q(x))$, then $\Delta \models \exists x.(p(x) \wedge q(x))$.

(j) If $\Gamma \models (\varphi \Rightarrow \psi)$ and $\Delta \models (\psi \Rightarrow \varphi)$,then $\Gamma \cap \Delta \models (\varphi \Rightarrow \psi) \vee (\psi \Rightarrow \varphi)$.

2. Unification. (10 points)

(a) Assuming that x, y, z, v, w are variables, give a most general unifier for the expressions $p(t(x, y), r(z, z))$ and $p(t(t(w, z), v), w)$.

(b) What is the result of applying this unifier to these expressions.

(c) Is it possible for two expressions to have more than one most general unifier? If so, give a simple example. If not, give a one-sentence explanation.

3. Clausal Form. (10 points) Convert the following sentence to clausal form.

$$(\exists z.\forall y.p(z,y) \vee \forall y.\exists z.p(z,y)) \Rightarrow \exists z.\forall y.p(y,z)$$

5. Resolution. (20 points) Use resolution to show that the following set of clauses is unsatisfiable. Assume that *w*, *x*, *y*, and *z* are variables and *a* is an object constant.

$$\{\neg p(x, y), q(x, y, f(x,y))\}$$
$$\{\neg r(y, z), q(a, y, z)\}$$
$$\{r(y, z), \neg q(a, y, z)\}$$
$$\{p(x, g(x)), q(x, g(x), z)\}$$
$$\{\neg r(x,y), \neg q(x, w, z)\}$$

Note that this is a question about Resolution Theorem Proving. You will get zero points (nil, nada, rien, zip, nothing) unless you use resolution and/or factoring on each step.

5. Model Building. (10 points) Consider the following sentence.

$$\forall x.(\neg p(x) \vee q(x)) \Leftrightarrow \neg \forall x.(p(x) \wedge q(x)).$$

(a) Give an interpretation that satisfies this sentence,

(b) Is the sentence valid? If so, write "valid". If not, give an interpretation that falsifies it.

In your interpretation(s), use {john, paul, mary} as the universe of discourse.

6. Herbrand Models. (10 points) One popular version of the Herbrand Theorem states that a set of equality-free clauses is satisfiable if and only if it has a Herbrand model. If the word "clauses" is changed to "first-order sentences", does the theorem still hold? If so, explain why. If not, give a counterexample.

7. Theory Completeness. (20 points) A universal language is a first-order language without functions, explicit quantifiers, or equality. Free variables are universally quantified. For example, $p(x) \Rightarrow p(x)$ is equivalent to $\forall x.(p(x) \Rightarrow p(x))$ Now, consider a universal language with just one unary relation constant p and two object constants a and b. Which of the following sentences logically entail a theory that is complete for all sentences in this language? For each case, write "complete" or "incomplete".

(a) $p(a) \wedge p(b)$

(b) $p(a) \wedge \neg p(b)$

(c) $(p(a) \vee p(b)) \wedge (p(a) \vee \neg p(b)) \wedge (\neg p(a) \vee p(b)) \wedge (\neg p(a) \vee \neg p(b))$

(d) $p(x)$

(d) $p(x) \Rightarrow p(x)$

(e) $p(x) \Rightarrow \neg p(x)$

(f) $p(a) \wedge \neg p(a)$

(g) Where completeness is concerned, sentence (b) has an interesting property that differentiates from the other sentences. What is that property?

2007 Comprehensive Examination Solutions
Logic

1. Logic. (20 points)
(a) True
(b) False
(c) True
(d) False
(e) False
(f) True
(g) False
(h) False
(i) False
(j) True

2. Unification. (10 points)
(a) $\{w \leftarrow r(z, z), x \leftarrow t(r(z, z), z), y \leftarrow v\}$
(b) $p(t(t(r(z, z), z), v), r(z, z))$
(c) $\{x \leftarrow y\}$ and $\{y \leftarrow x\}$ are most general unifiers of $p(x)$ and $p(y)$.

3. Clausal Form. (10 points)
$\{\neg p(z, f(z)), p(y, b)\}$
$\{\neg p(z, a), p(y, b)\}$

4. Resolution. (10 points)

1. $\{\neg p(x, y), q(x, y, f(x,y))\}$	Premise
2. $\{\neg r(y, z), q(a, y, z)\}$	Premise
3. $\{r(y, z), \neg q(a, y, z)\}$	Premise
4. $\{p(x, g(x)), q(x, g(x), z)\}$	Premise
5. $\{\neg r(x,y), \neg q(x, w, z)\}$	Premise
6. $\{\neg q(a, x, y), \neg q(x, w, z)\}$	3, 5
7. $\{q(x, g(x),f(x, g(x))), q(x, g(x), z)\}$	1, 4
8. $\{\neg q(g(a), w, z)\}$	6, 7 (factoring 7)
9. $\{\}$	7, 8 (factoring 7)

5. Model Building. (10 points)
(a) $p^i = \{\}$
(b) $p^i = \{\text{john}\}$ and $q^i = \{\text{john}\}$

6. Herbrand Models. (10 points) The following sentence is satisfiable but has no Herbrand model.

$$r(a) \wedge \exists x.\neg r(x)$$

7. Theory Completeness. (20 points)
(a) Incomplete
(b) Incomplete
(c) Complete
(d) Complete
(e) Incomplete
(f) Complete
(g) Complete
(h) It is not complete but it is maximal in that every sentence in the universal language is either logically entailed or inconsistent.

Comprehensive Exam: Networks: (60 points)
Closed Book: Fall 2007

Prof. David R. Cheriton

October 31, 2007

1. (15 points total) *End to end*

   (a) (7 points) Define the so-called "end-to-end principle" as applied to the Internet and provide an example of how it has been applied.

   (b) (5 points) Britney Spears goes into rehab, blaming her troubles on the presence of "middleboxes" such as web caches, load-balancers, firewalls, VPN gateways in the Internet which she claims compromise the end-to-end principle. Describe to what degree she is right (if any) and wrong, if so. (You can predicate your comments on assumptions of what these various boxes do, if you are not familiar with their operation.)

   (c) (3 points) There is a known major risk to high winds coming up when fighting forest fires in steep terrain. In the summer of 1994 in Glenwood Springs, Colo., 13 firefighter died tragically when the wind came in a steep canyon in which they were fighting a forest fire (mirroring a similar trajedy in 1949). In the book "Fire on the Mountain," John N. Maclean documents how the firefighters did not receive a revised weather report because of "bureaucratic bungling" at the Bureau of Lands and Mines. How would an "end-to-end" firefighter operate?

2. (15 points total) *Transport Protocol Design*

   (a) (8 points) Describe for each of: a) slow start b) fast retransmit c) AIMD (additive-increase-multiplicative-decrease) what it is, how it works and why it is compelling to include in TCP. (If you do not recognize these terms, describe how TCP supports congestion control in the Internet.)

(b) (7 points) TCP performs a 3-way handshake on connection setup and connection teardown. Describe the purpose of each, how it accomplishes that, and what bad things could happen if you went to less mechanism, e.g. 2-way message exchange.

3. (15 points total) *Network Routing*

   (a) (7 points) Describe how an IP router handles an incoming packet, focusing how it determines which port to send it out (if any) and what changes it makes to the packet (if any).

   (b) (8 points) Compare and contrast the three basic routing techniques, namely: flooding, distance-vector and link-state.

4. (15 points total) *Ethernet*

   (a) (5 points) Describe how CSMA-CD (carrier-sensor multiple access-collision detection) provides good throughput with minimal delay in a 10 Mbps Ethernet, being as quantitative as you can. Recall that a minimum Ethernet packet is 64 bytes so a minimum-size packet is roughly 50 microseconds and the speed of light is one foot per nanosecond (but the signal propagates at closer to 50 percent of that speed.)

   (b) (5 points) Hillary Clinton, after hearing how successful CSMA-CD was in the original Ethernet, proposes to legislate its use as "universal healthcare" for all networks, including high-speed, low-speed, wireless, satellite, etc. Describe the issues that CSM-CD runs into as you change the speed of the network and go to other technologies such as wireless, switched, etc., again being quantitative.

   (c) (5 points) Describe how an Ethernet packet manages to find the destination port to which it is addressed in a switched Ethernet network. That is, does the Ethernet switch run a layer 2 routing protocol or what. Also, point out 3 key challenges in the way this facility is implemented.

The End

Comprehensive Exam: Networks: Answers (60 points) Closed Book: Fall 2007

Prof. David R. Cheriton

October 31, 2007

Note: these answers are longer and more details than expected for passing the exam. We expect answers to demonstrate a secure understanding of the core focus of each question, not provide detailed/complete explanations.

1. (15 points total) *End to end*

   (a) (7 points) Define the so-called "end-to-end principle" as applied to the Internet and provide an example of how it has been applied.
   Answer: The canonical example is file transfer over the network (think FTP) - although each router along the path from file source to destination could perform a variety of checksums (for instance, to validate packet integrity or that no router along the path has accidentally flipped a bit), this does not protect against disk read/write errors. As such, it is necessary for there to be an "end-to-end" checksum of the file after it has been written to disk, so all of the checksums in the intermediary stages are duplicating effort. On the other hand, if file transfer takes a long time and errors in the medium are common, it may take excessively long to transmit successfully - thus incremental checksums may be merited as a performance optimization.

   (b) (5 points) Britney Spears goes into rehab, blaming her troubles on the presence of "middleboxes" such as web caches, load-balancers, firewalls, VPN gateways in the Internet which she claims compromise the end-to-end principle. Describe to what degree she is right (if any) and wrong, if so. (You can predicate your comments on assumptions of what these various boxes do, if you are not familiar with their operation.)

Answer: Web caches, when implemented correctly, are simply a performance optimization and thus do not violate e2e

Loadbalancers interfere with e2e to some extent by making it impossible for sources to determine which destination their traffic will arrive at. However, because in typical use cases they are controlled by one of the endpoints, it would be hard to claim that this is too flagrant a violation of the spirit of the e2e argument if one considers the larger definition of "endpoint" as content source rather than simply the host at the end of the line

Firewalls, when controlled by a user (think windows firewall) behave from an e2e perspective much like loadbalancers - it's perhaps a technical violation but because it's controlled by the "real end" of the communications channel who could choose a different behavior if desirable, this seems reasonable. When implemented to keep users in check, however, as in many corporate environments, they absolutely violate e2e because they can prevent users who happen to know their traffic is safe from communicating if the firewall decides for some reason that the traffic is not safe.

VPN gateways seem to violate e2e in the sense that they maintain state about connections and drop packets which do not authenticate. However, from an end-device standpoint (when implemented well) they should simply seem like one of many hops through the network, and it would be difficult to claim that an end-device is harmed by dropping packets which would not authenticate upon arrival anyway - this is akin to an ethernet device dropping packets whose CRC does not validate. The maintenance of state is more troubling, since it suggests that should the VPN gateway go away and come back the existing end-to-end connection would be forever terminated - but the short-lived nature of most information flow today suggests that a web browser would simply retry after the VPN recovered.

NAT boxes recomputing the TCP checksum are really a violation of E2E, if you view TCP as providing end-to-end reliability.

(c) (3 points) There is a known major risk to high winds coming up when fighting forest fires in steep terrain. In the summer of 1994 in Glenwood Springs, Colo., 13 firefighter died tragically when the wind came in a steep canyon in which they were fighting a forest fire (mirroring a similar trajedy in 1949). In the book "Fire on the Mountain," John N. Maclean documents how the firefighters

did not receive a revised weather report because of "bureaucratic bungling" at the Bureau of Lands and Mines. How would an "end-to-end" firefighter operate?

Answer: An end-to-end firefighter would explicitly ask for periodic weather reports and decide for himself whether or not it was safe to proceed. Such a weather report would clearly need to include a timestamp so that the firefighter could determine whether the information was current enough. He/she would get out of the area immediately at any time a reliable up-to-date weather report was not available.

2. (15 points total) *Transport Protocol Design*

   (a) (8 points) Describe for each of: a) slow start b) fast retransmit c) AIMD (additive-increase-multiplicative-decrease) what it is, how it works and why it is compelling to include in TCP. (If you do not recognize these terms, describe how TCP supports congestion control in the Internet.)

   Answer:

   Slow start - when a TCP first starts, increase the window faster than it would normally increase (multiplicatively instead of additively) - this allows a TCP session to rapidly fill the pipe in cases where there is no congestion, and reduces the amount of time "wasted" well below the potential bandwidth especially for short connections (of which there are many on the modern internet)

   Fast retransmit: If a single packet is dropped (which can be sensed by receipt of the following packet), trigger the sender to resend that packet by re-acking the packet before it. The sender resends the last unack'ed packet when receiving duplicate acks. If the majority of drops are single-packet drops and reordering is uncommon (which is a reasonable assumption), this provides much more rapid recovery from packet loss than waiting for timeouts.

   AIMD: Increase the number of packets in flight slowly (additively) and reduce it quickly (multiplicatively). For each window transmitted, increase window size by 1. For each packet which is dropped, cut the window size in half. This allows TCP to always eventually fill the pipe (optimize bandwidth) while also backing away quickly when there is contention for bandwidth.

(b) (7 points) TCP performs a 3-way handshake on connection setup and connection teardown. Describe the purpose of each, how it accomplishes that, and what bad things could happen if you went to less mechanism, e.g. 2-way message exchange.

Answer:

Connection setup - SYN, SYN/ACK, ACK. Needs to match up sequence numbers (specified in SYN and SYN/ACK) and also reserve resources to deal with the connection on each side. A two-way handshake can allow a connection to be setup by an old spurious SYN packet in the network. A two-way handshake for connection setup can also allow denial-of-service attacks (it's easy to open lots of connections from arbitrary addresses by synthesizing packets, if you don't need to be able to receive anything at those addresses).

connection teardown - FIN, FIN/ACK, ACK(?). Bidirectional communication implies that each side must announce"no more data to send", and they both also need to hear confirmation. If the confirmation is dropped, then the FIN or FIN/ACK can be resent without harm, but if the FIN portion is dropped the other side of the connection is going to be sitting waiting for data forever. Thus 3 messages are necessary.

3. (15 points total) *Network Routing*

(a) (7 points) Describe how an IP router handles an incoming packet, focusing how it determines which port to send it out (if any) and what changes it makes to the packet (if any).

Answer: IP router receives a packet because the packet is destined for the ethernet address of one of its network interfaces. It then uses the destination IP address encoded in the packet (as well as, possibly, other information including IP source address and incoming interface) to perform a lookup in its local routing table to choose a next-hop ethernet address and outgoing interface, looking for the shortest matching prefix. The routing table entry typically specifies an interface/port for point-to-point links, or, for multi-point links, an interface or port plus information on how to find the destination party (such as by using ARP to find the MAC for some particular IP address). Default-free routers that find no matching entry in their forwarding table may generate an ICMP "destination unreachable" packet and send it to

the original packet's source address. It then rewrites the ethernet header so that the packet appears to be sourced from the outgoing interface and destined for the next-hop address specified. It also decrements the TTL of the packet (discarding if too low) as an easy and guaranteed way of preventing routing loops. An IP router typically looks up the destination IP address in the packet's IP header in its forwarding table,

In parallel, a router typically verifies the IP checksum of the packet, checks various IP header options (source-routing, record-route, etc), checks the packet TTL field, and decrements it. Perhaps the router also interprets the TOS field in the packet to place it in different output queues.

More complex router policies can examine arbitrary packet state (TCP ports being a simple and common example) and keep state from one packet to the next.

(b) (8 points) Compare and contrast the three basic routing techniques, namely: flooding, distance-vector and link-state.

Answer: Flooding is highly reliable and simple but has poor scaling properties.

Distance-vector protocols are relatively simple but can take a long time to converge and may not scale that well to large networks either. It can be difficult to drop a route from the routing table.

Link-state protocols propagate full knowledge of all links in the network and can converge faster than distance-vector protocols.

4. (15 points total) *Ethernet*

(a) (5 points) Describe how CSMA-CD (carrier-sensor multiple access-collision detection) provides good throughput with minimal delay in a 10 Mbps Ethernet, being as quantitative as you can. Recall that a minimum Ethernet packet is 64 bytes so a minimum-size packet is roughly 50 microseconds and the speed of light is one foot per nanosecond (but the signal propagates at closer to 50 percent of that speed.)

Answer: When the Ethernet segment is not in use, as determined by listening or sensing the network, CSMA-CD stations start transmitting packets right away, avoiding any latency associated with network scheduling. However, if two stations transmit at the same time, a collision occurs and is detected, allowing the

stations to quickly retry rather than relying on higher-level recovery. Any node that observes a collision "jams" the network to announce this fact. Ethernet requires a minimum 64-byte packet size to ensure that the collision is observed by all nodes. At 10Mbps, 64 bytes take 51.2 microseconds to transmit, and with signal propagation of about c/2, travels quite far, 7.5km. If P is the worst-case propagation from one of the network to the other. In the worst-case, another station starts transmission at time P after the first transmitting station starts, so the latter only detects after 2P. This means that, the network diameter can be roughly at most 3km for the protocol to work correctly.

After a collision, senders wait for a random period of time, and try again. Senders use an exponential backoff algorithm, and eventually drop their packets, to avoid livelock.

(b) (5 points) Hillary Clinton, after hearing how successful CSMA-CD was in the original Ethernet, proposes to legislate its use as "universal healthcare" for all networks, including high-speed, low-speed, wireless, satellite, etc. Describe the issues that CSM-CD runs into as you change the speed of the network and go to other technologies such as wireless, switched, etc., again being quantitative.

Answer: CSMA-CD explicitly assumes that everyone on the network can notice when senders collide, which is valid on a wired system, but not for wireless where two nodes may send at the same time and be unable to hear one another but a node in between could see collisions. High-speed and long-distance networks such as gigabit, and satellite, have a significant network diameter measured in transmission bytes. This translates into a very long minimum packet size needed to ensure collision detection. For instance, gigabit with 64-byte minimum-size packets can have a physical diameter of 75 meters. To use gigabit within a 7.5-km city requires a minimum 6400 byte packet size, which would likely waste a lot of bandwidth with small packets and lots of padding. Similarly, satellite networks have very large distances that make it prohibitively expensive to detect collisions after the fact.

With wireless networks, another problem comes up, namely the fact that not all wireless nodes are guaranteed to be reachable from each other, and there isn't necessarily any symmetry. Two far-away nodes on opposite ends of a node in the middle can be

both transmitting at the same time, not hearing each other, and more over, the middle node may not be able to communicate with anyone at all if those other nodes don't hear the middle node's transmissions and back off.

(c) (5 points) Describe how an Ethernet packet manages to find the destination port to which it is addressed in a switched Ethernet network. That is, does the Ethernet switch run a layer 2 routing protocol or what. Also, point out 3 key challenges in the way this facility is implemented.

Answer: Ethernet switches map ethernet addresses to output ports. When a packet is received on port P, the switch associates its source address with port P. Any subsequent messages destined for that address are sent out port P. If a packet arrives for an address not in the list, the switch sends it out every port, i.e. flooding. Thus, messages are always delivered and, if one assumes most communication is bidirectional, the switch broadcasts only when absolutely necessary. 1) Routing loops are a real danger when switches broadcast packets. 2) Devices which move from port to port (for instance, when a computer is moved) cause packets to be sent in the wrong direction. 3) If the switch is plugged into another switch there may be many many addresses associated with each port which suggests a potential memory allocation issue.

The MAC address table mechanism is also very insecure, allowing anyone to capture packets destined for other hosts and break their communication by changing MAC table entries.

The End

Numerical Analysis Comprehensive Exam

Fall 2007

Use the attached sheets to write your answers.

1. Let $A \in \mathbb{R}^{m\times n}$ where $m \geq n$, and let $b \in \mathbb{R}^m$. Consider the least squares problem

$$\min_{x\in\mathbb{R}^n} \|Ax - b\|_2 \tag{1}$$

   (a) (3 points) Let $\phi(x) = \frac{1}{2}\|Ax-b\|_2^2$. A minimizer of $\phi(x)$, $x \in \mathbb{R}^n$ satisfies the equation $\nabla\phi(x) = 0$. Use this to derive a linear equation whose solution solves (1). How would you solve such a system?

   (b) (2 points) Under what conditions is a solution of (1) unique?

   (c) Let $A = U\Sigma V^T$ be the singular value decomposition of A, and let $\text{rank}(A) = r$. Let U and V be written in terms of their columns as

$$U = [u_1, \ldots, u_m], \quad V = [v_1, \ldots, v_n]$$

   Let

$$\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_r, \underbrace{0, \ldots, 0}_{n-r})$$

      i. (4 points) The minimum norm solution x_{LS} is the solution of (1) such that $\|x_{LS}\|_2 \leq \|x\|_2$ for any solution x. Show that the minimum norm solution of (1) is given by

$$x_{LS} = \sum_{i=1}^{r} \frac{u_i^T b}{\sigma_i} v_i$$

      ii. (2 points) Show that the associated residual has norm

$$\|Ax_{LS} - b\|_2^2 = \sum_{i=r+1}^{m} (u_i^T b_i)^2$$

2. Consider the scalar initial value problem $\dot{y} = \lambda y$, $x \in \mathbb{R}$, $y(x_0) = y_0$, $\lambda < 0$.

   (a) (1 point) Give the update rule for the numerical solution using backward Euler.

   (b) (1 point) Give the update rule for the numerical solution using trapezoidal rule.

   (c) (2 points) Describe the qualitative behavior of both update rules as Δt becomes very large.

3. Consider the equation $m\ddot{x} + c\dot{x} + kx = 0$ where $m > 0$, $c > 0$, $k > 0$, $c^2 > 4km$.

   (a) (2 points) Convert the equation to a system of the form $\dot{u} = Au$ and find the eigenvalues of the matrix.

   (b) (2 points) What choice of Δt will make forward Euler absolutely stable?

   (c) (2 points) What choice of Δt will make trapezoidal rule absolutely stable?

4. (4 points) Describe an algorithm for finding $z \in \mathbb{R}^n$, $z \neq 0$ such that $Uz = 0$ where $U \in \mathbb{R}^{n\times n}$ is upper triangular and $u_{nn} = 0$, $u_{11}, \ldots, u_{n-1,n-1} \neq 0$.

5. Consider vectors x, $y \in \mathbb{R}^n$ such that $\|x\|_2 = \|y\|_2 \neq 0$.

   (a) (3 points) Find a reflection matrix A such that $Ax = y$.

   (b) (2 points) Describe how you would compute Az for a given vector z in $O(n)$ time.

(Write your answers here.)

(Write your answers here.)

(Write your answers here.)

(Write your answers here.)

Numerical Analysis Comprehensive Exam, Fall 2007

I (a) $\phi(x) = \frac{1}{2}\|Ax-b\|_2^2 = \frac{1}{2}(Ax-b)^T(Ax-b)$

$= \frac{1}{2}(x^TA^TAx - x^TA^Tb - b^TAx + b^Tb)$

$= \frac{1}{2}(x^TA^TAx - 2b^TAx + b^Tb)$ (since $x^TA^Tb = b^TAx$)

The necessary condition for a minimizer of $\phi(x)$ is $\nabla\phi(x) = 0$. Differentiating ϕ,

$$0 = \nabla\phi(x) = \frac{1}{2}(2x^TA^TA - 2b^TA) \Rightarrow \boxed{A^TAx = A^Tb} \quad (\circledast)$$

This linear system gives the normal equations for the least squares problem (1). Any solution of (1) must necessarily satisfy the normal equations. Conversely, a solution of $\circledast$ satisfies (1).
The linear system $\circledast$ is symmetric, positive semi definite. For smaller systems, it could be efficiently solved using a Cholesky factorization and forward / backward substitution. For large, sparse systems, the method of conjugate gradients could be used. Note that (1) can also be solved without forming the normal equations.

I (b) The solution of (1) is unique iff A has full column rank. If A is rank deficient, $\exists z \neq 0$ s.t. $Az = 0$, so that if x is a solution of (1), $x+z$ is also a solution. If A has full column rank, the solution of $\circledast$ is unique (A^TA is non-singular), and since $\circledast$ is a necessary condition, the solution of (1) is unique.

II (c) (i) $\|Ax-b\|_2^2 = \|U\Sigma V^Tx - b\|_2^2 = \|\Sigma V^Tx - U^Tb\|_2^2$

where the last equality holds because $\|\cdot\|_2$ is invariant under orthogonal transformations. Next,

$$\|\Sigma V^Tx - U^Tb\|_2^2 = \sum_{i=1}^{n}(\sigma_i v_i^Tx - u_i^Tb)^2 + \sum_{i=n+1}^{m}(-u_i^Tb)^2$$

$$= \sum_{i=1}^{r}(\sigma_i v_i^Tx - u_i^Tb)^2 + \sum_{i=r+1}^{m}(-u_i^Tb)^2, \quad \text{using the fact that } \operatorname{rank}(A) = r$$

Thus, in order to minimize the 2-norm of the residual, we must choose x s.t.

$$v_i^Tx = \frac{u_i^Tb}{\sigma_i}, \quad i = 1, \cdots, r \qquad (\star\star)$$

Since $\{v_k\}_{k=1}^n$ form an orthonormal basis, we have $x = \sum_{k=1}^{n} v_k(v_k^Tx)$.
$(\star\star)$ gives the value of v_k^Tx for $k = 1, \cdots, r$, and to minimize the 2-norm of, we thus choose $v_k^Tx = 0$ for $k = r+1, \cdots, n$. Thus x_{LS} is given by

$$x_{LS} = \sum_{k=1}^{n} v_k(v_k^Tx) = \sum_{k=1}^{r} v_k(v_k^Tx)$$

$$\boxed{x_{LS} = \sum_{k=1}^{r} v_k \frac{u_k^Tb}{\sigma_k}}$$

[1] (c) (ii) From part (i), we have

$$\|Ax_{LS} - b\|_2^2 = \|\Sigma V^T x_{LS} - U^T b\|_2^2 = \sum_{i=1}^{r} \left(\sigma_i v_i^T x_{LS} - u_i^T b\right)^2 + \sum_{i=r+1}^{m} \left(-u_i^T b\right)^2$$

x_{LS} is chosen so that the first term in the sum vanishes. The second term does not depend on x. Thus

$$\|r\|_2^2 = \|Ax_{LS} - b\|_2^2 = \sum_{i=r+1}^{m} \left(u_i^T b\right)^2$$

2 (a) backward Euler

$$\frac{y^{n+1} - y^n}{\Delta t} = \lambda y^{n+1}$$

$$\Rightarrow \quad (1 - \lambda \Delta t) y^{n+1} = y^n$$

$$\Rightarrow \quad y^{n+1} = (1 - \lambda \Delta t)^{-1} y^n$$

(b) trapezoidal rule

$$\frac{y^{n+1} - y^n}{\Delta t} = \frac{1}{2} \lambda y^n + \frac{1}{2} \lambda y^{n+1}$$

$$\Rightarrow \quad y^{n+1} = y^n + \frac{\Delta t}{2} \lambda y^n + \frac{\Delta t}{2} \lambda y^{n+1}$$

$$\Rightarrow \quad \left(1 - \frac{\Delta t}{2} \lambda\right) y^{n+1} = \left(1 + \frac{\Delta t}{2} \lambda\right) y^n$$

$$\Rightarrow \quad y^{n+1} = \left(1 - \frac{\Delta t}{2} \lambda\right)^{-1} \left(1 + \frac{\Delta t}{2} \lambda\right) y^n$$

(c) as $\Delta t \to \infty$, $(1 - \lambda \Delta t)^{-1} \to 0$, so the backward Euler update gives $y^{n+1} \to 0$, i.e., a steady state solution of 0.
For trapezoidal rule, as $\Delta t \to \infty$, $y^{n+1} \to \frac{\Delta t \lambda / 2}{-\Delta t \lambda / 2} y^n = -y^n$, so the solution will oscillate, flipping sign in each iteration.

3 (a) We can rewrite this as a first order system, $u' = Au$, where

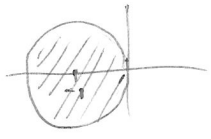
$$u = \begin{pmatrix} x \\ \dot{x} \end{pmatrix}, \quad A = \begin{pmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{c}{m} \end{pmatrix}.$$

The eigenvalues of the matrix A are the roots of the characteristic polynomial

$$|A - \lambda I| = \begin{vmatrix} -\lambda & 1 \\ -\frac{k}{m} & -\frac{c}{m} - \lambda \end{vmatrix} = \lambda\left(\lambda + \frac{c}{m}\right) + \frac{k}{m} = \lambda^2 + \frac{c}{m}\lambda + \frac{k}{m}.$$

Using the quadratic formula, these are $\lambda = -\frac{c}{m} \pm \sqrt{\left(\frac{c}{m}\right)^2 - 4\frac{k}{m}}$.

(b) The update rule for forward Euler is $y^{n+1} = (1 + \Delta t \lambda) y^n$, so for absolute stability we require $|1 + \Delta t \lambda| \leq 1 \quad \forall \lambda$, or $|\Delta t \lambda - (-1)| \leq 1$, so that $\Delta t \lambda$ lies in the unit circle in the complex plane centered at -1. Thus Δt must be chosen so that this holds for all eigenvalues of A. Note that λ above has nonpositive real part.



(c) For trapezoidal rule to be absolutely stable, we need

$$\frac{\left|1 + \frac{\Delta t}{2} \lambda\right|}{\left|1 - \frac{\Delta t}{2} \lambda\right|} \leq 1.$$

Since λ has nonpositive real part, this inequality is satisfied. (This can be easily seen as the numerator is the distance of $\frac{\Delta t}{2}\lambda$ to -1, and the denominator is the distance of $\frac{\Delta t}{2}\lambda$ to 1.) Thus trapez. rule is absolutely stable for any choice of Δt.

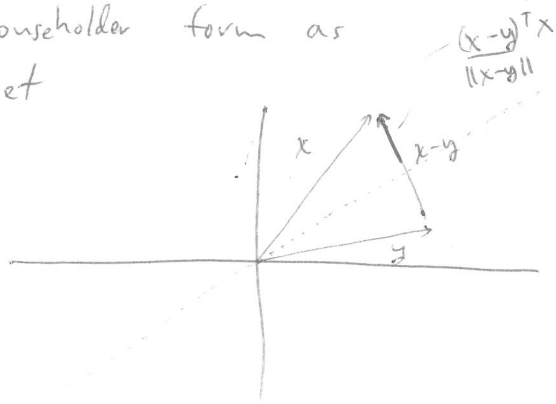
4 We require $\sum_{j=1}^{n} u_{ij} z_j = \sum_{j=i}^{n} u_{ij} z_j = 0$ for each $i \in 1 \cdots n$.

This is satisfied unconditionally for $i = n$ since $u_{nn} = 0$, so set $z_n = 1$. The other components of z can be computed in reverse order via

$$z_i = \frac{-1}{u_{ii}} \sum_{j=i+1}^{n} u_{ij} z_j$$

since $u_{ii} \neq 0$ for $i < n$ and z_j, $j > i$ are known when z_i is computed.

5 (a) Writing the unknown reflection in Householder form as $A = I - 2uu^T$ for $|u| = 1$, we get



$$Ax = y$$

$$x - 2u(u^T x) = y$$

$$u = \frac{x - y}{2u^T x} \quad (*)$$

Since $|u| = 1$, it suffices to set $u = \frac{x-y}{\|x-y\|}$ unless $x = y$, in which case u can be any vector orthogonal to $x = y$. $(*)$ holds since

$$2u^T x = 2\frac{x^T x - y^T x}{(x^T x - 2y^T x + y^T y)^{1/2}} = \frac{x^T x - 2y^T x + y^T y}{(x^T x - 2y^T x + y^T y)^{1/2}} = \|x - y\|$$

if $\|x\| = \|y\|$.

(b) It suffices to compute Az as

$$Az = (I - 2uu^T)z = z - 2u(u^T z)$$

$$= z - (2u^T z)u$$

Comprehensive Exam: Programming Languages Autumn 2007

This is a 60-minute closed-book exam and the point total for all questions is 60.

All of the intended answers may be written within the space provided. (*Do not use a separate blue book.*) Succinct answers that do not include irrelevant observations are preferred. You may use the back of the preceding page for scratch work. If you use the back side of a page to write part of your answer, be sure to mark your answer clearly.

The following is a statement of the Stanford University Honor Code:

A. *The Honor Code is an undertaking of the students, individually and collectively:*

   (1) *that they will not give or receive aid in examinations; that they will not give or receive unpermitted aid in class work, in the preparation of reports, or in any other work that is to be used by the instructor as the basis of grading;*

   (2) *that they will do their share and take an active part in seeing to it that others as well as themselves uphold the spirit and letter of the Honor Code.*

B. *The faculty on its part manifests its confidence in the honor of its students by refraining from proctoring examinations and from taking unusual and unreasonable precautions to prevent the forms of dishonesty mentioned above. The faculty will also avoid, as far as practicable, academic procedures that create temptations to violate the Honor Code.*

C. *While the faculty alone has the right and obligation to set academic requirements, the students and faculty will work together to establish optimal conditions for honorable academic work.*

By writing my "magic number" below, I certify that I acknowledge and accept the Honor Code.

_____

(Number)

Prob	# 1	# 2	# 3	# 4	# 5	Total
Score						
Max	15	10	11	12	12	60

1. (*15 points*) ..... Short Answer

Answer each question in a few words or phrases.

(a) (*3 points*) Can a language that does not allow explicit deallocation and uses a correct implementation of garbage collection have dangling pointers? Justify your answer.

(b) (*3 points*) What is a closure and what problem does it solve?

(c) (*3 points*) Explain the difference between subtyping and inheritance (in at most two sentences).

(d) (*3 points*) Assume that `Rectangle` is a subtype of `Shape`, written `Rectangle` <: `Shape`. Which of the following subtype relationships hold in principle?

i. (`Shape` → `Rectangle`) <: (`Rectangle` → `Rectangle`)

ii. (`Rectangle` → `Shape`) <: (`Rectangle` → `Rectangle`)

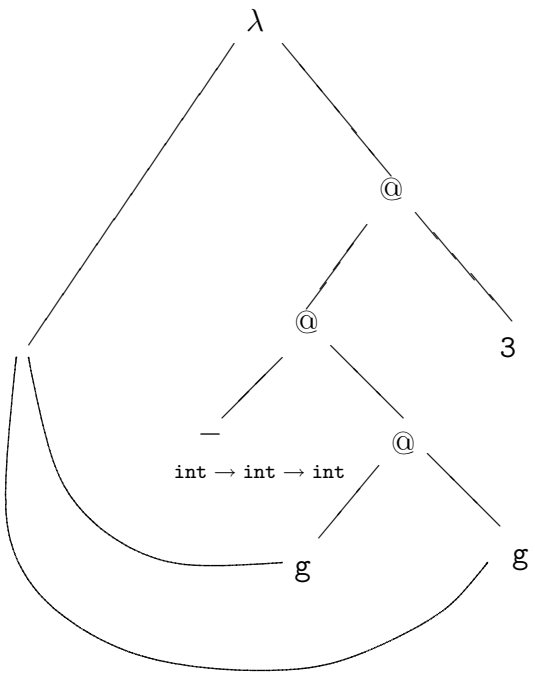
(e) (*3 points*) Why do static fields of a Java class have to be initialized when the class is loaded? Why can't we initialize static fields when the program starts?

2. (*10 points*) ..... Type Inference on Parse Graph

Use the parse graph below to follow the steps of the ML type inference algorithm on the function declaration

```
fun f(g) = g(g) - 3;
```

Write the type associated with each node of the graph, as the type inference algorithm proceeds from the bottom of the graph up towards the root. What is the output of the type checker?



3. (*11 points*) ..... Parameter passing comparison

For the following Algol-like program, write the number printed by running the program under each of the listed parameter passing mechanisms.

In *pass-by-value-result*, also called call-by-value-result and copy-in/copy-out, parameters are passed by value, with an added twist. More specifically, suppose a function `f` with a pass-by-value-result parameter `u` is called with actual parameter `v`. The activation record for `f` will contain a location for formal parameter `u` that is initialized to the R-value of `v`. Within the body of `f`, the identifier `u` is treated as an assignable variable. On return from the call to `f`, the actual parameter `v` is assigned the R-value of `u`.

```
begin
   integer i;

   procedure pass ( x, y );
      integer x, y;  // types of the formal parameters
      begin
         x := x + 3;
         y := x + 5;
         x := y;
         i := i + 7
      end

   i := 1;
   pass (i, i);
   print i
end
```

(a) (*3 points*) pass-by-value

(b) (*4 points*) pass-by-reference

(c) (*4 points*) pass-by-value/result

4. (*12 points*) ..... Phantom Members

A C++ class may have virtual members that may be redefined in derived classes. However, there is no way to "undefine" a virtual (or non-virtual) member. Suppose we extend C++ by adding another kind of member, called a *phantom* member, that is treated as virtual, but only defined in derived classes if an explicit definition is given. In other words, a "phantom" function is not inherited unless its name is listed in the derived class. For example, if we have two classes

```
class A {
...
public:
   phantom void f(){...}
   ...
};
class B : public A {
...
public:
   ... /* no definition of f */
};
```

then `f` would appear in the `vtbl` for `A` objects and, if `x` is an `A` object, `x.f()` would be allowed. However, if `f` is not declared in `B`, then `f` might not need to appear in the `vtbl` for `B` objects and, if `x` is a `B` object, `x.f()` would not be allowed.

(a) (*8 points*) Are phantom members consistent with the design of C++, or is there some general property of the way the language is designed and implemented that would be destroyed? If so, explain what this property is, why it is important, and why it is destroyed.

(b) (*4 points*) JavaScript does allow a method of an object to be removed. Explain why this is consistent with the design goals and implementation mechanisms for JavaScript, referring to your answer to part (a) as appropriate.

5. (*12 points*) ..... Race Conditions

The program on the following page contains two classes: `RaceInducer` creates a field `counter` that is the source of data races, and field `counter` is an object of class `DoubleCounter`, which supports methods `incrementBoth` and `getDifference`. For each of the following questions, justify your answer *briefly.*

(a) (*3 points*) If `incrementBoth` and `getDifference` were never allowed to execute at the same time, what would this program print?

(b) (*3 points*) In the program, `dif_value` in `run` is always either 0 or 1. Why is that? Why can't the difference exceed 1 or become negative?

(c) (*2 points*) One way to ensure that data races do not occur would be to insert synchronization primitives. For example, declaring

```
public synchronized int getDifference() {...}
public int incrementBoth() {...}
```

would prevent two threads from executing in method `getDifference` at the same time. Is this enough to ensure that `getDifference` always returns 0?

(d) (*2 points*) Is the following declaration

```
public int getDifference() {...}
public synchronized int incrementBoth() {...}
```

sufficient to ensure that `getDifference` always returns 0?

(e) (*2 points*) If the following declaration is used,

```
public synchronized int getDifference() {...}
public synchronized int incrementBoth() {...}
```

what will the output be? Explain.

```
class RaceInducer extends Thread {
//  this object is shared between all instances of this class
    static DoubleCounter counter;
    static volatile boolean done = false;

    public void run() {
        try {
            for (int i = 0; i < 1000; i++) {
                if (i % 60 == 0) {
                    // insert line break
                    System.out.println();
                }
                int dif_value = counter.getDifference();
                // prints either a '+' or a '-'
                System.out.print("+-".charAt(dif_value));
                sleep(20); // suspends the current thread
            }
            done = true;
        } catch (InterruptedException e) {
            return;
        }
    }

    // entry point into the program
    public static void main(String[] x) {
        Thread ri = new RaceInducer();
        counter = new DoubleCounter();
        try {
            ri.start();     // starts a new thread and calls run()
            while (!done) {
                counter.incrementBoth();
                sleep(30);  // suspends the current thread
            }
            ri.join();
        } catch (InterruptedException e) {
            return;
        }
    }
}

/**
 * Shared data structure.
 * */
class DoubleCounter {
    protected int x = 0, y = 0;

    public int getDifference() {
        return x - y;
    }

    public void incrementBoth() throws InterruptedException {
        x++;
        Thread.sleep(9);
        y++;
    }
}
```

Comprehensive Exam: Programming Languages Autumn 2007

1. (*15 points*) ..... Short Answer

   Answer each question in a few words or phrases.

   (a) (*3 points*) Can a language that does not allow explicit deallocation and uses a correct implementation of garbage collection have dangling pointers? Justify your answer.

   Answer: This answer is no. The only time a piece of heap memory is freed by a correct garbage collector is when the program no longer can access it. Since dangling pointers occur when memory is freed when a program may still access it, there are no dangling pointers as a result of correct garbage collection.

   (b) (*3 points*) What is a closure and what problem does it solve?

   Answer: A closure is a pair consisting of a pointer to code and a pointer to an activation record, used to represent a function and its lexical environment. Closures are used to preserve static scope when a function is passed to another function, or returned from a function.

   (c) (*3 points*) Explain the difference between subtyping and inheritance (in at most two sentences).

   Answer: Subtyping is a relation between interfaces and inheritance is a relation between implementations.

   (d) (*3 points*) Assume that `Rectangle` is a subtype of `Shape`, written `Rectangle` <: `Shape`. Which of the following subtype relationships hold in principle?

   i. (`Shape` → `Rectangle`) <: (`Rectangle` → `Rectangle`)

   ii. (`Rectangle` → `Shape`) <: (`Rectangle` → `Rectangle`)

   Answer: The first subtype relationship is correct, because function types are contravariant in the argument type.

   (e) (*3 points*) Why do static fields of a Java class have to be initialized when the class is loaded? Why can't we initialize static fields when the program starts?

   Answer: Static fields are initialized when a class is loaded because this is the first point at which initialization is possible. It is important to initialize fields before they are accessed, and static fields may be used by static methods before any instances (objects) of the class are created.

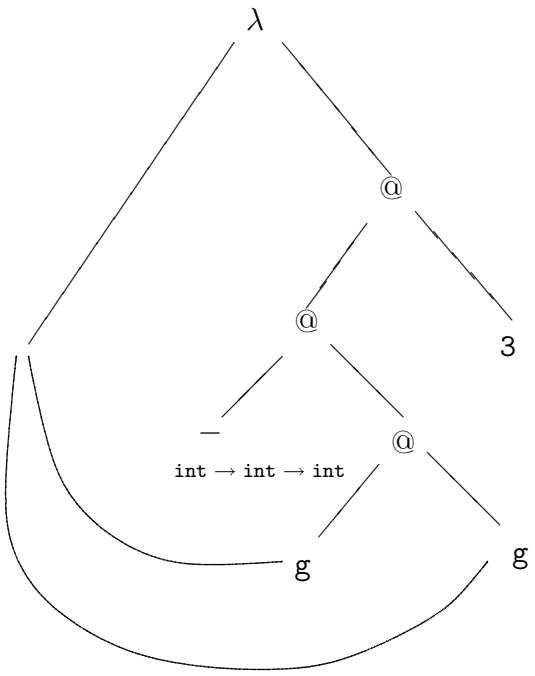
2. (*10 points*) ..... Type Inference on Parse Graph

   Use the parse graph below to follow the steps of the ML type inference algorithm on the function declaration

   ```
   fun f(g) = g(g) - 3;
   ```

   Write the type associated with each node of the graph, as the type inference algorithm proceeds from the bottom of the graph up towards the root. What is the output of the type checker?

   Answer: The function is not typeable in ML; the type checker reports *error.*



Answer: The type algorithm determines that the type a of g must satisfy some constraint of the form $a = a \to b$. This form of constraint cannot be solved by substituting type expressions for type variables.

3. (*11 points*) ..... Parameter passing comparison

For the following Algol-like program, write the number printed by running the program under each of the listed parameter passing mechanisms.

In *pass-by-value-result*, also called call-by-value-result and copy-in/copy-out, parameters are passed by value, with an added twist. More specifically, suppose a function `f` with a pass-by-value-result parameter `u` is called with actual parameter `v`. The activation record for `f` will contain a location for formal parameter `u` that is initialized to the R-value of `v`. Within the body of `f`, the identifier `u` is treated as an assignable variable. On return from the call to `f`, the actual parameter `v` is assigned the R-value of `u`.

```
begin
   integer i;

   procedure pass ( x, y );
      integer x, y;  // types of the formal parameters
      begin
         x := x + 3;
         y := x + 5;
         x := y;
         i := i + 7
      end

   i := 1;
   pass (i, i);
   print i
end
```

(a) (*3 points*) pass-by-value

Answer: 8

(b) (*4 points*) pass-by-reference
Answer: 16

(c) (*4 points*) pass-by-value/result
Answer: 9

4. (*12 points*) ..... Phantom Members

A C++ class may have virtual members that may be redefined in derived classes. However, there is no way to "undefine" a virtual (or non-virtual) member. Suppose we extend C++ by adding another kind of member, called a *phantom* member, that is treated as virtual, but only defined in derived classes if an explicit definition is given. In other words, a "phantom" function is not inherited unless its name is listed in the derived class. For example, if we have two classes

```
class A {
...
public:
   phantom void f(){...}
   ...
};
class B : public A {
...
public:
   ... /* no definition of f */
};
```

then `f` would appear in the `vtbl` for `A` objects and, if `x` is an `A` object, `x.f()` would be allowed. However, if `f` is not declared in `B`, then `f` might not need to appear in the `vtbl` for `B` objects and, if `x` is a `B` object, `x.f()` would not be allowed.

(a) (*8 points*) Are phantom members consistent with the design of C++, or is there some general property of the way the language is designed and implemented that would be destroyed? If so, explain what this property is, why it is important, and why it is destroyed.
Answer: If we add phantom functions to the language, then it will be possible to use public inheritance and *not* get a subtype. This is because we might choose to leave a phantom function undefined in a subclass. This breaks a property of C++that an initial segment of the `vtable` of a derived class matches the form of the `vtable` its base class. However, it might be possible for for C++ to allow phantom members and recognize that some derived classes do not result in subtypes.

(b) (*4 points*) JavaScript does allow a method of an object to be removed. Explain why this is consistent with the design goals and implementation mechanisms for JavaScript, referring to your answer to part (a) as appropriate.
Answer: JavaScript uses run-time type checking, so does not rely on static typing or subtyping for implementation efficiency or correctness.

5. (*12 points*) ..... Race Conditions

The program on the following page contains two classes: `RaceInducer` creates a field `counter` that is the source of data races, and field `counter` is an object of class `DoubleCounter`, which supports methods `incrementBoth` and `getDifference`. For each of the following questions, justify your answer *briefly.*

(a) (*3 points*) If `incrementBoth` and `getDifference` were never allowed to execute at the same time, what would this program print?

Answer: The difference will always be 0, so a sequence of + signs will be printed, with a newline every 60 + signes.

(b) (*3 points*) In the program, `dif_value` in `run` is always either 0 or 1. Why is that? Why can't the difference exceed 1 or become negative?

Answer: Method `incrementBoth` is the only method that changes the values of `x` and `y`, and only one thread calls `incrementBoth` in this program. Therefore, every time that `x` is incremented, `y` is incremented next before `x` can be incremented again. The only reason the difference would be non-zero is if `getDifference` is allowed to execute *while* `incrementBoth` is running. Therefore, the difference is either 0 or 1.

(c) (*2 points*) One way to ensure that data races do not occur would be to insert synchronization primitives. For example, declaring

```
public synchronized int getDifference() {...}
public int incrementBoth() {...}
```

would prevent two threads from executing in method `getDifference` at the same time. Is this enough to ensure that `getDifference` always returns 0?

Answer: No. Only one thread will be executing `getDifference`, however, another thread can run while `incrementBoth` is running, leading to the same kind of behavior as before.

(d) (*2 points*) Is the following declaration

```
public int getDifference() {...}
public synchronized int incrementBoth() {...}
```

sufficient to ensure that `getDifference` always returns 0?

Answer: No. Again, only one thread will be executing `incrementBoth`, however, another thread can run `getDifference` at the same time. As a result, while the first thread is running, the second will get access to partially modified values, leading to the same exact problem as before.

(e) (*2 points*) If the following declaration is used,

```
public synchronized int getDifference() {...}
public synchronized int incrementBoth() {...}
```

what will the output be? Explain.

Answer: A bunch of + signs will be printed. This version places a lock on the only `DoubleCounter` object in the program ensures that only one thread will be running either of these methods at once. In other words, these two methods will be executed sequentially.

```
class RaceInducer extends Thread {
//  this object is shared between all instances of this class
    static DoubleCounter counter;
    static volatile boolean done = false;

    public void run() {
        try {
            for (int i = 0; i < 1000; i++) {
                if (i % 60 == 0) {
                    // insert line break
                    System.out.println();
                }
                int dif_value = counter.getDifference();
                // prints either a '+' or a '-'
                System.out.print("+-".charAt(dif_value));
                sleep(20); // suspends the current thread
            }
            done = true;
        } catch (InterruptedException e) {
            return;
        }
    }

    // entry point into the program
    public static void main(String[] x) {
        Thread ri = new RaceInducer();
        counter = new DoubleCounter();
        try {
            ri.start();     // starts a new thread and calls run()
            while (!done) {
                counter.incrementBoth();
                sleep(30);  // suspends the current thread
            }
            ri.join();
        } catch (InterruptedException e) {
            return;
        }
    }
}

/**
 * Shared data structure.
 * */
class DoubleCounter {
    protected int x = 0, y = 0;

    public int getDifference() {
        return x - y;
    }

    public void incrementBoth() throws InterruptedException {
        x++;
        Thread.sleep(9);
        y++;
    }
}
```

Comprehensive Exam — Systems software

Fall 2007

October 30, 2007

Answer the questions below in your blue book. *You may skip 10 points worth of questions.* These questions are 5 points each.

1. You are using round robin and run the same job mix on a faster machine but get a non-linear improvement in completion time. What happened?
2. Under memory pressure OS Y's page replacement algorithm reuses clean pages before dirty ones. Why would it do this? What is the downside of doing so?
3. Why might two processes that run correctly with paging deadlock when paging is disabled?
4. Can a first fit allocator ever have less fragmentation than best fit? If not, why not? If so, give an example.
5. Is it possible for a CPU scheduler with a 100 millisecond time slice to spend over half its time in the OS context switch code? Assume context switching takes 1 millisecond. Justify your answer.
6. Explain how reference bits can be emulated on an architecture that has a normal paging system but lacks hardware support for reference bits.
7. Explain why an OS can save less state on a system call trap than a page fault trap.
8. On a system with a TLB what does the OS have to do after revoking a page from a process?

These two questions are 10 points each.

1. Assume the common Unix file system interface, in particular, that you have a way to *non-atomically* write file data, that you have `sync()`, and that rename is atomic. Explain how to overwrite a file `A` with new contents such that any crash will result in A having either the old or new contents. Give the sequence of calls you would do for this.
2. Describe the optimal spinlock blocking algorithm for a multiprocessor assuming you have perfect knowledge. Give a spin-before-blocking algorithm that will always be within 2x of optimal.