Stanford University
Computer Science Department
CS 343 midterm
Spring 2011

May 1, 2012

!!!!!! SKIP 20 POINTS WORTH OF QUESTIONS. !!!!!

This is an open-book exam. You have 75 minutes. Write all of your answers directly on the paper. Make your answers as concise as possible. Sentence fragments ok.

NOTE: We will take off points if a correct answer also includes incorrect or irrelevant information. (I.e., don't put in everything you know in hopes of saying the correct buzzword.)

Question	Score
1-5 (25 points)	
6 (15 points)	
7 (30 points)	
8-9 (25 points)	
total (75 points)	

Stanford University Honor Code

In accordance with both the letter and the spirit of the Honor Code, I did not cheat on this exam nor will I assist someone else cheating.

Name and Stanford ID:

Signature:

Answer the following short answer questions in a sentence or two, *say why your answer holds*. (5 points each).

1. Atom goes to great lengths to avoid modifying the addresses in the data segment. However, in Figure 4 the program text is at a different location. Why did the authors do this? How do they mitigate any problem that could arise from this relocation?

2. Which papers have a sentence that roughly says: "single entry, multiple exit" in them?

3. The transmeta hardware provided a branch instruction with two targets. Pick a figure in one of the papers we've read that shows a good place to use this instruction and state why.

4. Give three real problems other systems handle that ATOM does not and say how to implement these.

Problem 6: Static Checking (15 points)

1. (10 points) There are many "return-owner" routines that return the sole reference to an object, which the caller must track; losing this "owning" reference is a leak error, since it means the object has been lost. The `malloc` routine is an example:

```
int *contrived(void) {
        int *p = malloc(sizeof *p);
        if(!p)
                return 0;   // not an error: p = null.
        if(foo() < 0)
                return 0;   // error: lost p!
        return p;           // not an error: returned ref.
}
```

   Give pseudo-code for static checker that emits an error when a owned pointer has been lost (as in the example). Assume that a owned pointer has been safely handled if it is (1) returned, (2) assigned to another variable, (3) or passed to any function.

2. (5 points) Using ideas from the belief analysis lecture: Give an intuitive sketch of how to statistically infer which routines return owned pointers using your checker.

Problem 7: No new ideas (30 points) Valgrind, Dynamo, Pin, and TraceMonkey all do roughly the same thing: take the text of a program and run it, trying to get speed by translating some portion to (possibly additionally optimized) executable code.

(10 points each): Compare how they do the following. Please be very concrete!

1. Explain what "linking" is in the context of a translation cache. For each system: do they do it, why or why not? How do they handle eviction?

2. Compare how they optimize indirect jumps (e.g., a control flow to the address held in a register).

3. ATOM, Pin, Valgrind, Dynamo, Pin: List one clear advantage each system has over the others.

Problem 8: Purify (15 points) You have the following pieces of code. For each, say whether Purify will (1) always catch the error, (2) might catch it, or (3) will miss it. Make sure to justify your answer.

1.
```
int *p = my_alloc_routine(100);
p[1000] = 5;
```

2.
```
struct foo {
        int x[10];
        int bar;

};
struct foo *x = malloc(sizeof *x);
x->x[10] = 1;
```

3.
```
void foo(){
        void *p = malloc(10);
}
```

4.
```
p = malloc(10);
...
free(p);
...
 q = malloc(10);
*p = 10;
...
```

Problem 9: BAM (10 points)

1. (5 points) Explain why synchronous exceptions are hard and which systems (out of Valgrind, Dynamo, Pin and ATOM) had to make an explicit effort to handle them.

2. (5 points) Sketch how to use Transmeta's hardware to handle exceptions and how this would make one of the above systems better.