CpSc 418 **Midterm** October 22, 2013

100 points

Time for the exam: 80 minutes.

Open book: anything printed on paper may be brought to the exam and used during the exam. This includes the textbook, other books, printed copies of the lecture slides, lecture notes, homework and solutions, and any other material that a student chooses to bring.

Calculators are allowed: no restriction on programmability or graphing. There are a few simple calculations needed in the exam, a calculator will be handy, but the fancy features will not make a difference.

No communication devices: That's right. You may not use your cell phone for voice, text, web-surfing, or any other purpose. Likewise, the use of computers, iPods, etc. is not permitted during the exam.

Test books: I have included space with each question for you to write your answers. You may use a test booklet if you need more space.

Good luck!

CpSc 418 **Midterm** October 22, 2013

Graded out of **100 points** (102 points are possible)

0. (**2** points)

   (a) Your name: _____

   (b) Your student number: _____

1. **Erlang** (20 points) Consider the following code for computing the Fibonacci numbers:

```
fib(1) -> 0;
fib(2) -> 1;
fib(N) when is_integer(N), N > 2 -> fib(N-1) + fib(N-2).
```

   I tried it, and got the results shown below:

`N`	`fib(N)`	time
1	0	0.28μs
2	1	0.28μs
3	1	0.34μs
4	2	0.41μs
5	3	1.7μs
10	34	3.4μs
15	377	35.2μs
20	4181	389.3μs
25	46368	4.3ms
30	514229	47.6ms
35	5702887	528.9ms
40	63245986	5.8s

   (a) (**5 points**) Show that for $N > 10$, the runtime is roughly proportional to `fib(N)`. Give a one sentence explanation for why this is the case.

(b) (**5 points**) Fill-in the blanks to complete the more efficient calculation of the Fibonacci numbers below:

```
fib2(1) -> 0;
fib2(N) when is_integer(N), N > 1 ->

   hd(fib2(N, _______)).

fib2(2, AB) -> AB;

fib2(N, _______) ->

   ____________________________________________.
```

(c) (**10 points**) Consider the function `score` defined below:

```
score(PPid, N) ->
   case N of
      0 -> receive V when V >= 0 -> score(PPid, 4-V)      end;
      1 -> receive V when V >= 0 -> 10                    end;
      3 -> receive V when V >= 0 -> score(PPid, V*V)      end;
      5 -> receive V when V >= 0 -> score(PPid, V*V+2)    end;
      7 -> score(PPid, 10);
      9 -> receive V when V >= 0 -> score(PPid, N-V)      end;
      _ when N =< 10 -> PPid !  N
   end.
```

Fill-in values for the three send expressions in the code below to complete a function, `grader`, that when executed will print your score for this problem:

```
grader() ->
   MyPid = self(),

   CPid = spawn(fun() -> score(MyPid, 0) end),

   CPid ! __________

   CPid ! __________

   CPid ! __________
   io:format("your score on this problem is ~b~n",
             [receive S -> S after 1000 -> 0 end]).
```

2. **Performance** (30 points) Short answer. Each answer should be one to three sentences. Points may be taken off for answers longer than 100 words.

   (a) (**5 points**) What is *super-linear speed-up*?

   Name one common cause/explanation for super-linear speed-up.

   (b) (**5 points**) What is *communication overhead*? Define the term *and* give a one or two sentence example.

   (c) (**5 points**) What is *computation overhead*? Define the term *and* give a one or two sentence example.

(d) (**5 points**) What is *Amdahl's Law*? Give the mathematical formula *and* a short, English explanation of what it suggests about parallel computing.

(e) (**5 points**) Write a short (less than five lines) code fragment, and point out a RAW dependency. Explain why it is a dependency.

(f) (**5 points**) Write a short (less than five lines) code fragment – you may use the same code as for question 2e – and point out a WAR hazard. Give a short explanation of how the hazard can be removed by renaming.

3. Network Topologies. (25 points) Consider a messaging passing machine with 64 processors. This problem examines the routing latency for various topologies. We will assume a latency of one time unit for each network link that the message traverses. All network links are bi-directional.

Example: if the processors are arranged as an 8×8 mesh, then processor i (for $0 \leq i < 64$) will be at the mesh location $(i \div 8, i \bmod 8)$ where $i \div j$ denotes integer division (truncating), and $i \bmod j$ indicates the remainder when i is divid by j (like `%` in C/Java or `rem` in Erlang. For example, processor 17 will be at location $(2, 1)$ and processor 42 will be at location $(5, 2)$. The minimum latency to send a message from processor 17 to processor 42 is 4 time units (three hops along one dimension, and one hop along the other).

(a) (**5 points**) If the processors are arranged as a ring, we can number the processors $0 \ldots 63$, so that processor i has a bi-directional link to processors $(i+1) \bmod 64$ and $(i-1) \bmod 64$. What is the minimum number of routing hops to send a message from processor 17 to processor 42?

(b) If the processors are arranged as a $4 \times 4 \times 4$ mesh, then processor i will be at location $(i \div 16, (i \div 4) \bmod 4, i \bmod 4)$.

i. (**5 points**) What are the coordinates for processors 17 and 42?

ii. (**5 points**) What is the minimum time (shortest path) to send a message from processor 17 to processor 42?

(c) If the processors are arranged as a 6-dimensional hypercube, then processor i has location $((i \div 32),\ (i \div 16) \bmod 2,\ (i \div 8) \bmod 2,\ (i \div 4) \bmod 2,\ (i \div 2) \bmod 2,\ i \bmod 2)$.

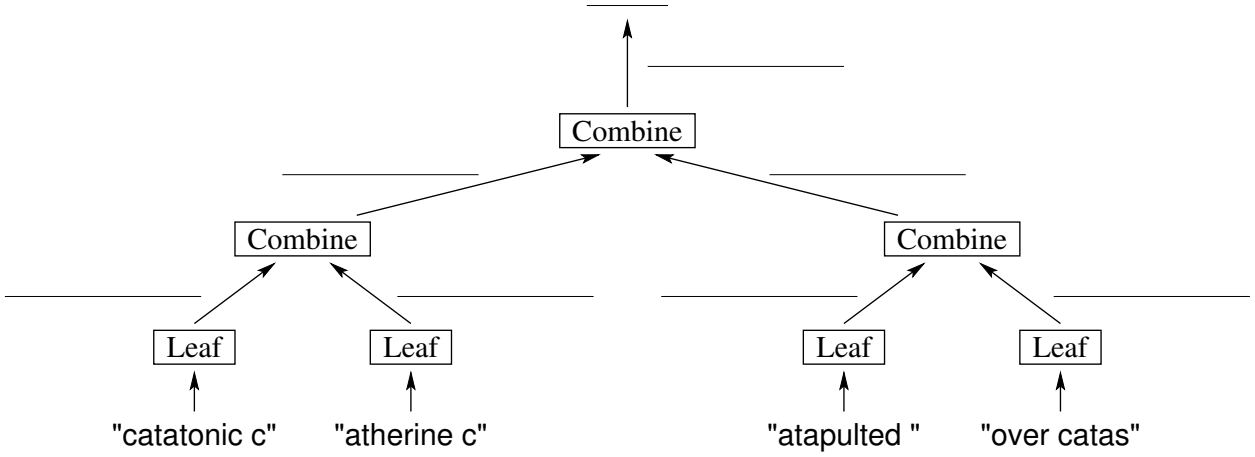
i. (**5 points**) What are the coordinates for processors 17 and 42?

ii. (**5 points**) What is the minimum time (shortest path) to send a message from processor 17 to processor 42?

4. **Counting Cats** (25 points) Given a string, `S`, `cats(S)` is the number of occurrences of the substring `"cat"` in S. For example, if

```
S =    "catatonic catherine catapulted over catastrophic cataracts in"
    ++ "her catamaran while caterwauling scathing acatelectic scat"
    ++ "scattering cattle from catalpas and catering to categories of"
    ++ "caterpillars".
```

(a) (5 points) Complete the figure of the tree below to show how counting cats can be done in parallel. In other words, fill in the blank lines with the values that should be there.



(b) (10 points) Write a stub for the `Leaf` function:

- What are the parameters to the function?

- What does it return?

- **Briefly** describe how it could be implemented.

- Give **one** example of a call to `Leaf` and what it returns.

(c) (10 points) Write a stub for the `Combine` function:

- What are the parameters to the function?

- What does it return?

- **Briefly** describe how it could be implemented.

- Give **one** example of a call to `Combine` and what it returns.