15-150 Assignment 03

Jack Kasbeer
jkasbeer@andrew.cmu.edu
Section K
September 22, 2015

2: Zippidy Doo Da

---

1. `zip : (int list, int list) -> (int, int) list`

2. ML runtime systems says "`Warning: match nonexhaustive`". This means that the function definition does not cover every case for the arguments. It probably expects us to cover the cases of when one of the arguments is an empty list and the other is not.

3. The function does not satisfy Specification 1 because it must be the case that the lists are the same length in the current implementation. It does however satisfy Specification 2 for this exact reason. It also satisfies Specification 3 even though the ensures statement isn't very informative and is somewhat redundant.

4. (THM): For all lists A_1, B_1, A_2, B_2 of the same type, if

$$length(A_1) = length(A_2) \text{ and } length(B_1) = length(B_2)$$

then

$$zip(A_1@B_1, A_2@B_2) = zip(A_1, A_2)\,@\,zip(B_1, B_2)$$

Proof: By induction on the length of A_1, B_1, A_2, B_2.

Proof. Let A_1, B_1, A_2, B_2 be lists of the same type and assume that

$$length(A_1) = length(A_2) \text{ and } length(B_1) = length(B_2)$$

Base case: When $length\,A_1, B_1, A_2$, or $B_2 = 0$, then $length\,A_1@B_1$ and $length\,A_2@B_2$ are both 0 (by the definition of `zip`).
Let $P(n) \Rightarrow zip(A_1@B_1, A_2@B_2) = zip(A_1, A_2)\,@\,zip(B_1, B_2)$ for some length n. Assume that $P(k)$ holds for some k. WWTS $P(k+1)$ holds in order to prove $P(n)\,\forall n \geq 0$
We start by examining $P(k)$... This means $zip(A_1@B_1, A_2@B_2) = zip(A_1, A_2)\,@\,zip(B_1, B_2)$ where $k = length\,A_1, B_1, A_2, B_2$.

□

3: Counting

---

1. $A_1@B_1$ has length $n + m$ and performs n cons operations (first bullet point). Similarly, $A_2@B_2$ has length $n + m$ and performs n cons operations (first bullet point). Thus, $A_1@B_1$ and $A_2@B_2$ both have length $n+m$, so $zip(A_1@B_1, A_2@B_2)$ evaluates to a list of length $n+m$ and performs $n + m$ cons operations (second bullet point). Hence, $zip(A_1@B_1, A_2@B_2)$, in total, performs $n + n + (n + m) = 3n + m$ cons operations.

2. A_1 and A_2 both have length n, so let $C = zip(A_1, A_2)$, and notice it evaluates to a list of length n and performs n cons operations (second bullet point). Similarly, B_1 and B_2 both have length m, so let $D = zip(B_1, B_2)$, and notice it evaluates to a list of length m and performs m cons operations (second bullet point). Thus, ignoring internal cons operations, $C\,@\,D$ evaluates to a list value of length $n+m$ and performs n cons operations. Hence, in total, $zip(A_1, A_2)\,@\,zip(B_1, B_2)$ performs $n+m+n = 2n+m$ cons operations.

4: Hits and Strips

1. Task 4.1 is in `hw03.sml`
2. Task 4.2 is in `hw02.sml`

5: Look and Say

1. Task 5.1 is in `hw03.sml`
2. Task 5.2 is in `hw03.sml`

6: Prefix Sums

1. $W_{add\_to\_each}(n)$
   $W_{add\_to\_each}(0) = c_0$
   $W_{add\_to\_each}(n) = c_0 + c_1 + W_{add\_to\_each}(n-1)$
   $\Rightarrow W_{add\_to\_each}(n) = c_0 + nc_1,\ \forall n \geq 0$

   $W_{prefix\_sums}(n)$
   $W_{prefix\_sums}(0) = k_0$
   $W_{prefix\_sums}(1) = k_0 + W_{add\_to\_each}(1)$
   $W_{prefix\_sums}(2) = W_{prefix\_sums}(1) + W_{add\_to\_each}(2)$
   $W_{prefix\_sums}(n) = k_0 + W_{prefix\_sums}(n-1) + W_{add\_to\_each}(n)$
   $W_{prefix\_sums}(n) = k_0 + n(c_0 + nc_1)$
   $\Rightarrow W_{prefix\_sums}(n) = k_0 + nc_0 + n^2c_1,\ \forall n \geq 0$

2. $W_{add\_to\_each}(n) = c_0 + nc_1,\ \forall n \geq 0 \Rightarrow O(n)$
   $W_{prefix\_sums}(n) = k_0 + nc_0 + n^2c_1,\ \forall n \geq 0 \Rightarrow O(n^2)$

3. Show that the work to evaluate `fast_prefix_sums L`, when `length L = n`, is `O(n)`.
   $W_{fast\_prefix\_sums}(0) = W_{prefix\_sums\_helper}(0)$
   $W_{fast\_prefix\_sums}(1) = W_{prefix\_sums\_helper}(1)$

   There's no additional work in `fast_prefix_sums` other than the work done by `prefix_sums_helper`.

$W_{fast\_prefix\_sums}(n) = W_{prefix\_sums\_helper}(n)$
So we must find the work done by `prefix_sums_helper`.
$W_{prefix\_sums\_helper}(0) = c_0$
$W_{prefix\_sums\_helper}(1) = c_0 + W_{prefix\_sums\_helper}(0)$
$W_{prefix\_sums\_helper}(n) = c_0 + c_1 + W_{prefix\_sums\_helper}(n-1)$
$\Rightarrow W_{prefix\_sums\_helper}(n) = c_0 + nc_1$
$\therefore W_{fast\_prefix\_sums}(n) = c_0 + nc_1,\ \forall n \geq 0$
Hence, $W_{fast\_prefix\_sums}(n)$ is $O(n)$.

4. Task 6.4 is in `hw03.sml`

7: Sum Nights

1. Task 7.1 is in `hw03.sml`
2. Task 7.2 is in `hw03.sml`
3. Task 7.3 is in the `queue`