Homework 33

Brian Knotten, Brett Schreiber, Brian Falkenstein

November 13, 2018

21

Consider a sequential algorithm for this problem on a binary tree T as follows:

1. Let $i \leftarrow 1$.
2. Perform an Eulerian tour on the tree.
3. If the node is a leaf, label the node with i and increment i.

The result is a tree with each leaf node labelled in-order.
The outline for the parallel version of this algorithm is as follows:

1. Perform an Eulerian tour on T to return a $3n$ linked-list L.
2. Let the nodes in L representing a leaves in the tree (only one visit on the tour) be set to 1 and let their corresponding processor keep a pointer to its node.
3. Let all other nodes in L be set to 0.
4. Perform the linked-list parallel prefix algorithm on L.
5. Each processor corresponding to a leaf node in T looks at its pointer to the node in L to see a prefix sum. This number represents the number of the leaf in T in an in-order traversal.
6. Have each processor label their leaf node in T with this number.

This algorithm takes $O(\log(n))$ time, since performing an Eulerian tour takes a constant number of steps with n processors, and performing the parallel prefix sum algorithm takes $O(\log(n))$ steps. This algorithm is EREW, because the Eulerian tour algorithm is EREW, the parallel prefix algorithm is EREW, and each processor reads and writes exclusively to its corresponding nodes in the steps particular to this algorithm.

Formally, the algorithm is as follows:

Algorithm 1 EREW $O(\log(n))$ algorithm for In-Order labeling of leaf nodes with n processors.

```
Require: An $n$-sized binary tree $T$.
  $t_i \leftarrow$ the node in $T$ which $p_i$ points to.
  $t_p \leftarrow t_i$'s parent node.
  if $t_i$ is a leaf node then
      Allocate a copy of $t_i$ for creating $L$: $l_{i_1}$
      if $t_i$ is a left child then
          Link $l_{i_1} \rightarrow l_{p_2}$
      else if $t_i$ is a right child then
          Link $l_{i_1} \rightarrow l_{p_3}$
      end if
  else
      $t_l \leftarrow t_i$'s left child.
      $t_r \leftarrow t_i$'s right child.
      Allocate three copies of $t_i$ for creating $L$: $l_{i_1}$, $l_{i_2}$, $l_{i_3}$.
      Link $l_{i_1} \rightarrow l_{l_1}$
      Link $l_{i_2} \rightarrow l_{r_1}$
      if $t_i$ is a left child then
          Link $l_{i_3} \rightarrow l_{p_2}$
      else if $t_i$ is a right child then
          Link $l_{i_3} \rightarrow l_{p_3}$
      end if
  end if
  $L' \leftarrow$ the parallel prefix algorithm on $L$.
  if $t_i$ is a leaf node then
      Label $t_i$ with $l'_{i_1}$
  end if
```

23

As indicated by the hint, to solve the problem of evaluating a binary expression tree with the four standard arithmetic operations as nodes and integers as leaves it will suffice to find a collection of functions that is closed under the four standard operations with constants, composition, and contains the identity and constant functions and apply that class of functions to the algorithm presented in class.

Note that in class we used the class of functions of the form $ax + b$ (where a is 1 or -1, and b is any number) for addition and subtraction.

Now we present a new class of functions of the form $\frac{ax+b}{nx+m}$ and show that this class of functions satisfies the presented requirements:

1. Identity function: $\frac{1x+0}{0x+1} = \frac{x}{1} = x$

2. Constant function: $\frac{0x+k}{0x+1} = \frac{k}{1} = k$ where k is any constant

3. Closed under addition of a constant: $\frac{ax+b}{nx+m} + \ell = \frac{ax+b}{nx+m} + \frac{\ell(nx+m)}{nx+m} = \frac{ax+b+\ell nx+\ell m}{nx+m} = \frac{(a+\ell n)x+b+\ell m}{nx+m} = \frac{yx+z}{nx+m}$ where $y = a + \ell n$ and $z = b + \ell m$

4. Close under subtraction of a constant: $\frac{ax+b}{nx+m} - \ell = \frac{ax+b}{nx+m} + \frac{(-\ell)(nx+m)}{nx+m} = \frac{ax+b+(-\ell)nx+(-\ell)m}{nx+m} = \frac{(a+(-\ell)n)x+b+(-\ell)m}{nx+m} = \frac{yx+z}{nx+m}$ where $y = a + (-\ell)n$ and $z = b + (-\ell)m$

5. Closed under multiplication by a constant: $\frac{ax+b}{nx+m} \cdot c = \frac{cax+cb}{nx+m} = \frac{yx+z}{nx+m}$ where $y = ca$ and $z = cb$

6. Closed under division by a constant: $\frac{\frac{ax+b}{nx+m}}{d} = \frac{ax+b}{nx+m} \cdot \frac{1}{d} = \frac{ax+b}{dnx+dm} = \frac{ax+b}{yx+z}$ where $y = dn$ and $z = dm$

7. Closed under the division of a constant: $\frac{d}{\frac{ax+b}{nx+m}} = d \cdot \frac{nx+m}{ax+b} = \frac{dnx+dm}{ax+b} = \frac{yx+z}{ax+b}$ where $y = nx$ and $z = dm$

8. Closed under the composition of functions: Let $f(x) = \frac{ax+b}{nx+m}$ and let $g(x) = \frac{cx+d}{ex+f}$. Then $f(g(x)) = \frac{a(\frac{cx+d}{ex+f})+b}{n(\frac{cx+d}{ex+f})+m} = \frac{\frac{acx+ad}{ex+f}+b}{\frac{ncx+nd}{ex+f}+m} = \frac{\frac{acx+ad}{ex+f}+\frac{bex+bf}{ex+f}}{\frac{ncx+nd}{ex+f}+\frac{mex+mf}{ex+f}} = \frac{ex+f}{ex+f} \cdot \frac{(acx+ad)+(bex+bf)}{(ncx+nd)+(mex+mf)} = \frac{(ac+be)x+(ad+bf)}{(nc+me)x+(nd+mf)} = \frac{yx+z}{wx+v}$ where $y = (ac+be)$, $z = (ad+bf)$, $w = (nc+me)$, and $v = (nd+mf)$

We have shown that our class of functions is closed under the four standard operations with constants and composition, and contains the identity and constant functions. Therefore, our class of functions can be applied to the $O(log^2 n)$ algorithm given in class that applies "cuts" using these functions to every odd numbered left child that is a leaf or even numbered right child that is a leaf at each step. Because our operations can all be done in constant time, our functions can be applied to the class algorithm without changing the $O(log^2 n$ runtime on a CREW PRAM with n processors.