EE 382V: Parallel Algorithms **Summer 2017**

Lecture 6: Blelloch Scan

Lecturer: Vijay Garg *Scribe: Van Quach*

6.1 Parallel Scan

Scan is a simple and useful parallel building block to convert recurrences from sequential into parallel. 2 types of scan:

- Inclusive scan: last input element is included in the result

- Exclusive scan: last input element is included in the result

6.1.1 Hillis & Steele Scan

This algorithm is one of the most important building blocks for data-parallel computation.

Algorithm 1 Hillis & Steele scan

procedure PREFIX_SUM(A) ▷ This algorithm is using base-1 index
    if $n = 1$ **then**
        $C[1] = A[1]$
    end if
    for $i \leftarrow 1, \frac{n}{2}$ **do in parallel**
        $B[i] = A[2i-1] + A[2i]$
    end for
    $D = parallel\_prefix(B[1] \dots B[n])$
    for $i \leftarrow 1, n-1$ **do in parallel**
        if $i = 1$ **then**
            $C[i] = A[i]$
        else if even(i) **then**
            $C[i] = D[i/2]$
        else if odd(i) **then**
            $C[i] = D[i/2] + A[i]$
        end if
    end for
end procedure

A	1	2	3	4	5	6	7	8
B	3	7	11	15				
D	3	10	21	36				
C	1	3	6	10	15	21	28	36

Figure 6.1: Arrays' values

	$T(n)$	$W(n)$
$Hillis\&SteeleScan$	$\mathcal{O}(\log(n))$	$\mathcal{O}(n \times \log(n))$

Table 6.1: Time and Work

6.1.2 Blelloch Scan

Blelloch scan is also called 2-sweeps algorithm. The idea of Blelloch scan algorithm is do upward sweep followed by downward sweep. Blelloch scan is an exclusive scan. Upward sweep use 6.1 as a base equation:

$$scan(x) = sum(L) + sum(R) \tag{6.1}$$

Downward sweep is more complicated, and use 6.2, and 6.3 as base equations.

$$scan(L(x)) = scan(x) \tag{6.2}$$

$$scan(R(x)) = sum(L(x)) + scan(x) \tag{6.3}$$

Algorithm 2 Blelloch Scan

```
procedure PREFIX_SUM(A)
    for i ← 0, n − 1 do in parallel
        B[i] = A[i]
    end for
    for h ← 0, log(n) − 1 do                                  ▷ upward sweep
        for i ← 0, n − 1 in steps of 2^(h+1) do in parallel
            B[i + 2^(h+1) − 1] = B[i + 2^h − 1] + B[i + 2^(h+1) − 1]
        end for
    end for
    B[n − 1] = 0                                              ▷ Set root to 0
    for h ← log(n) − 1, 0 do                                  ▷ downward sweep
        left_value = B[i + 2^h − 1]
        B[i + 2^h − 1] = B[i + 2^(h+1) − 1]
        B[i + 2^(h+1) − 1] = left_value + B[i + 2^(h+1) − 1]
    end for
end procedure
```

	$T(n)$	$W(n)$
Sequential	$\mathcal{O}(n)$	$\mathcal{O}(n)$
Parallel	$\mathcal{O}(\log(n))$	$\mathcal{O}(n \times \log(n))$
Hillis&Steele	$\mathcal{O}(\log(n))$	$\mathcal{O}(n \times \log(n))$
Blelloch	$\mathcal{O}(\log(n))$	$\mathcal{O}(n)$

Figure 6.2: Summary of Time and Work