Parallel Recursion: Ladner-Fischer Parallel Prefix Sum

Greg Plaxton
Theory in Programming Practice, Fall 2005
Department of Computer Science
University of Texas at Austin

Prefix Sum

- Fix an associative binary operator $\oplus$ defined over some domain
  - Let 0 denote a left identity element of $\oplus$, i.e., assume that $0 \oplus x = x$ for all x in the domain of $\oplus$
- Throughout our discussion of parallel prefix, we consider only powerlists for which the elements are drawn from the domain of $\oplus$
- The parallel prefix problem is to compute the function that maps any given powerlist $p = \langle x_0 \; \ldots x_{n-1} \rangle$ to the powerlist

$$\langle x_0 \;\; (x_0 \oplus x_1) \;\; (x_0 \oplus x_1 \oplus x_2) \;\; \ldots \;\; (x_0 \oplus \ldots \oplus x_{n-1}) \rangle$$

- For the sake of brevity, we refer to this function as f in what follows

Ladner-Fischer Parallel Prefix Scheme

- If $n > 1$, apply $\oplus$ to successive pairs of elements to obtain the length-$n/2$ powerlist

$$p' = \langle (x_0 \oplus x_1)\ (x_2 \oplus x_3)\ \ldots\ (x_{n-2} \oplus x_{n-1}) \rangle$$

- Recursively compute the prefix sum of p' to obtain the length-$n/2$ powerlist

$$p'' = \langle (x_0 \oplus x_1)\ (x_0 \oplus x_1 \oplus x_2 \oplus x_3)\ \ldots\ (x_0 \oplus \ldots \oplus x_{n-1}) \rangle$$

  – The powerlist p'' contains the odd-indexed elements of $f(p)$

  – To get the even-indexed elements of $f(p)$, take the $\oplus$ of the powerlist obtained by shifting p'' to the right one position (and introducing a 0 in the first position) with

$$\langle x_0\ \ x_2\ \ x_4\ \ \ldots x_{n-2} \rangle$$

A Powerlist Formulation of the LF Scheme: Overview

- Definition of the $*$ operator
- The LF scheme revisited
- A powerlist specification of the prefix sum operation
- Derivation of the LF scheme

Definition of the $*$ Operator

- For any powerlist $p = \langle x_0 \; \ldots \; x_{n-1} \rangle$, we define p^* as the powerlist $\langle 0 \;\; x_0 \;\; x_1 \; \ldots \; x_{n-2} \rangle$
- Here is an inductive definition of p^*

$$
\begin{aligned}
\langle x \rangle^* &= \langle 0 \rangle \\
(p \bowtie q)^* &= q^* \bowtie p
\end{aligned}
$$

Remark: Some Properties of the $*$ Operator

- Property 1: $(p \oplus q)^* = p^* \oplus q^*$
  - This property may be proven by induction
  - The proof is left as an exercise
- Property 2: $(p \bowtie q)^{**} = p^* \bowtie q^*$
  - By the definition of $*$, $(p \bowtie q)^{**} = (q^* \bowtie p)^*$
  - Applying the definition of $*$ a second time yields the desired equation
- We will not need these particular properties in the proofs that follow

The LF Scheme Revisited

- Using the powerlist notation, we can write the LF scheme for computing the parallel prefix function f as follows

$$\begin{aligned} f(\langle x \rangle) &= \langle x \rangle \\ f(p \bowtie q) &= (t^* \oplus p) \bowtie t \text{ where } t = f(p \oplus q) \end{aligned}$$

- In what follows, we show how to derive the above equation from a powerlist-based specification of the prefix sum operation

Specification of Prefix Sum

- Consider the equation $q = q^* \oplus p$ in the given powerlist $p = \langle x_0 \ldots x_{n-1} \rangle$ and the unknown powerlist $q = \langle y_0 \; \ldots \; y_{n-1} \rangle$
- This equation has a unique solution in q
  - Note that $y_0 = 0 \oplus x_0 = x_0$
  - Thus $y_1 = y_0 \oplus x_1 = x_0 \oplus x_1$, so $y_2 = y_1 \oplus x_2 = x_0 \oplus x_1 \oplus x_2$, et cetera
  - In general, $y_i = x_0 \oplus x_1 \oplus \ldots \oplus x_i$, $0 \leq i < n$
  - In other words, the unique solution (in q) to the equation $q = q^* \oplus p$ is $f(p)$

Derivation of the LF Scheme

- We wish to derive an equation for $f(p \bowtie q)$, where p and q are equal-length powerlists
- Since $f(p \bowtie q)$ is a non-singleton powerlist, there is a unique way to write it in the form $r \bowtie t$
- Our plan is to solve for r and t in what follows
- By the result of the previous slide, $r \bowtie t = (r \bowtie t)^* \oplus (p \bowtie q)$
- By the definition of $*$, the latter expression is equal to $(t^* \bowtie r) \oplus (p \bowtie q)$
- Since $\oplus$ is a pointwise operator, $\oplus$ and $\bowtie$ commute and the previous expression can be rewritten as $(t^* \oplus p) \bowtie (r \oplus q)$

Derivation of the LF Scheme (continued)

- Thus far we have established that $r \bowtie t = (t^* \oplus p) \bowtie (r \oplus q)$
  - By unique deconstruction, $r = t^* \oplus p$ and $t = r \oplus q$
  - Hence $t = (t^* \oplus p) \oplus q = t^* \oplus (p \oplus q)$
  - Earlier we saw that the unique solution to this equation is $t = f(p \oplus q)$
- In summary, we have shown that $f(p \bowtie q)$ is equal to $(t^* \oplus p) \bowtie t$ where $t = f(p \oplus q)$
  - In effect we have derived the powerlist-based formulation of the LF scheme stated earlier

Sequential Complexity of the LF Scheme

- Let $T(n)$ denote the sequential running time of the LF scheme
- We obtain the recurrence $T(1) = O(1)$ and $T(n) = T(n/2) + O(n)$
- This recurrence solves to give $T(n) = O(n)$

Parallel Complexity of the LF Scheme

- Let $T(n)$ denote the parallel running time of the LF scheme using n processors
- We obtain the recurrence $T(1) = O(1)$ and $T(n) = T(n/2) + O(1)$
- This recurrence solves to give $T(n) = O(\log n)$
- In fact, the LF scheme can be used to compute prefix sum in $O(\log n)$ time using only $n/\log n$ processors
  - The overhead at the top level of recursion is $O(\log n)$, but it drops off by a factor of two at each successive level
  - This parallel algorithm is considered to be "work-efficient" because its processor-time product is equal (to within a constant factor) to the sequential time complexity