Parallelizing non-associative sequential reductions

THE UNIVERSITY of EDINBURGH informatics

by Federico Pizzuti

e-mail: pizzutifederico27@gmail.com
Supervisor: Christophe Dubach

EPSRC Centre for Doctoral Training in Pervasive Parallelism

Abstract

A great number of algorithms can be implemented in terms of reductions, a very common functional pattern. In most cases, reductions are easily parallelizable.

There are however many important reduction applications for which a parallel implementation cannot be directly derived from their natural functional form due to non-associative reduction operators.

We present a transformation capable of deriving a parallel implementation for many reductions with non-associative operators by leveraging the associativity of matrix multiplications defined over semirings.

A trivially parallel reduction: Summation of Numbers

The formula $\sum_0^n x_n$

can be expressed functionally as the reduction

reduce(+,[1,2,3,4,5,6,7,8])

$1 + 2 + 3 + 4 + 5 + 6 + 7 + 8$

Can we parallelize Polynomial Evaluation?

The formula $\sum_0^n x_n * k^n$

Equivalent to $x_0 \odot x_1 \odot \ldots \odot x_n$ where $a \odot b = k * a + b$

reduce($\odot$,[1,2,3,4,5,6,7,8]) $1 \odot 2 \odot 3 \odot 4 \odot 5 \odot 6 \odot 7 \odot 8$

Key insight: Matrix Multiplication is Associative

By rearranging the input [1,2,3,4,5,6,7,8] in this series of matrices

$$\begin{bmatrix} k & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} k & 2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} k & 3 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} k & 4 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} k & 5 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} k & 6 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} k & 7 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} k & 8 \\ 0 & 1 \end{bmatrix}$$

We can rewrite reduce($\odot$,[1,2,3,4,5,6,7,8]) = **x**

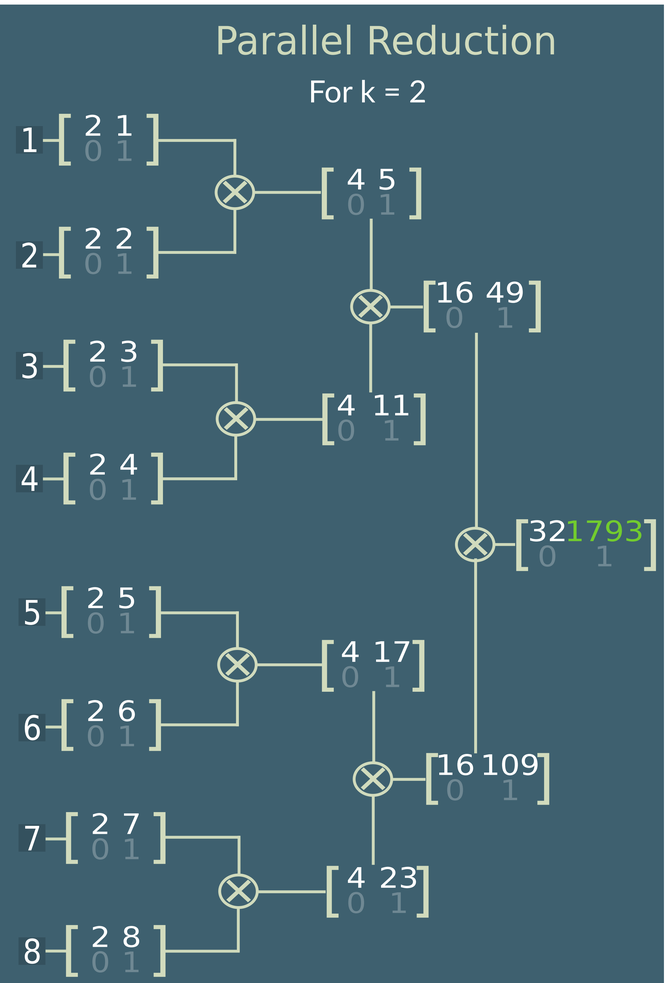
into reduce($\times$,[[k 1 / 0 1],[k 2 / 0 1],...]) = $\begin{bmatrix} a & \mathbf{x} \\ 0 & 1 \end{bmatrix}$

$\times$ is **associative***! We know how to parallelize this!

This rewrite is valid for other operators composed of **semiring operations**

Parallel Reduction

For k = 2



Associativity

$+$ **Associative** operator: the reduction steps can be executed in **any order**

$\odot$ **Non-associative** operator: the reduction steps must be executed **sequentially**

An operator $\oplus$ is associative if:

$$(a \oplus b) \oplus c = a \oplus (b \oplus c)$$

*Applicable for all Semiring Operations

$+$ Associative and communative operator

$*$ Associative operator, distributes over +

0 Neutral element for +

1 Neutral element for *

Why does this work?

Compare

$$\begin{bmatrix} k & a \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} k & b \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} k*k & k*a+b \\ 0 & 1 \end{bmatrix}$$

to

$$a \odot b = k * a + b$$

We can use matrix multiplication to compute $\odot$

Efficient implementation

We don't actually need to fully materialize the derived matrices, as many entries are just compile time constants

In this case for example, we just need to track the values of the first row

$$\begin{bmatrix} k & a \\ 0 & 1 \end{bmatrix} \Longrightarrow \begin{bmatrix} k & a \end{bmatrix}$$