Parallelizing non-associative sequential reductions

By Federico Pizzuti
Supervisor: Cristophe Dubach
Associative reductions are parallelizable

Summation of Numbers: \[ \sum_{0}^{n} x_n = x_1 + \ldots + x_n \]

\[ \text{reduce}(+, [1, 2, 3, 4, 5, 6, 7, 8]) \]

The operator + is associative, so this reduction is parallelizable
What about non-associative reductions?

Polynomial evaluation: \( \sum_{0}^{n} x_n \cdot k^n \)

Define operator: \( a \odot b = k \cdot a + b \)

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

The operator \( \odot \) is **not associative**, so this reduction must be executed **sequentially**

How to parallelize non-associative reductions?
Key insight: Matrix multiplication ($\times$) is associative*

We rewrite the reduction using $\oslash$ in terms of a reduction that uses $\times$

The derived reduction is then parallelizable

*This works for all operators expressed in terms of semiring operations, not just $+$ and $\cdot$. 
Rewrite reduction operator as matrix multiplication

Rearrange data as matrices

\[
\begin{bmatrix}
1, 2, 3, 4, 5, 6, 7, 8
\end{bmatrix}
\]

\[
\begin{bmatrix}
k_1 \\
0
\end{bmatrix} \times
\begin{bmatrix}
k_2 \\
0
\end{bmatrix} \times
\begin{bmatrix}
k_3 \\
0
\end{bmatrix} \times
\begin{bmatrix}
k_4 \\
0
\end{bmatrix} \times
\begin{bmatrix}
k_5 \\
0
\end{bmatrix} \times
\begin{bmatrix}
k_6 \\
0
\end{bmatrix} \times
\begin{bmatrix}
k_7 \\
0
\end{bmatrix} \times
\begin{bmatrix}
k_8 \\
0
\end{bmatrix}
\]

\[\text{reduce} (\odot, \begin{bmatrix} 1, 2, 3, 4, 5, 6, 7, 8 \end{bmatrix}) = x\]

\[\text{reduce} (\odot, \begin{bmatrix} k_1 \\ 0 \end{bmatrix}, \ldots, \begin{bmatrix} k_8 \\ 0 \end{bmatrix}) = \begin{bmatrix} a & x \\ 0 & 1 \end{bmatrix}\]