We introduce a generalised reduction over sequences. The function takes as parameters an associative reduction operator, an initial accumulator value, and a sequence of elements. The accumulator need not have the same type as the elements of the sequence. For instance, we can use a reduction to count the number of occurrences of a particular value in a sequence.

$$[S, T]$$

$$reduce : (S \times T \to S) \times S \times \operatorname{seq} T \to S$$

$$\begin{aligned}
&\forall f : S \times T \to S \bullet \forall s : S \bullet \forall ts : \operatorname{seq} T \bullet \\
&\quad reduce(f, s, ts) = \\
&\qquad \textbf{if } \# ts = 0 \\
&\qquad \textbf{then } s \\
&\qquad \textbf{else } reduce(f, f(s, head\ ts), tail\ ts)
\end{aligned}$$

This definition hints at a linear reduction, but an efficient implementation could perform a tree-like reduction on vector hardware.