This specification describes a tally system for first-past-the-post (FPTP) elections.

In an FPTP election, citizens are registered as voters and candidates. Every candidate is a voter, but not every voter is a candidate.

This an anonymous teller, which means that every voter is issued a token, and a voter votes for a token among the set of candidate tokens.

It is outside the scope of this specification to ensure the anonymity of tokens. It is also outside the scope of this specification to ensure that all voters are handed out a token, and no duplicate ballots occur.

$$[TOKEN]$$

$$\begin{array}{|l} voters : \mathbb{P}\ TOKEN \\ candidates : \mathbb{P}\ TOKEN \\ \hline candidates \subseteq voters \end{array}$$

$$\begin{array}{|l} ballot : voters \to candidates \end{array}$$

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.

$$\begin{array}{|l} [S, T] \\ \hline reduce : (S \times T \to S) \times S \times \mathrm{seq}\ T \to S \\ \hline \forall f : S \times T \to S \bullet \forall s : S \bullet \forall ts : \mathrm{seq}\ T \bullet \\ \quad reduce(f, s, ts) = \\ \quad\quad \textbf{if}\ \#ts = 0 \\ \quad\quad \textbf{then}\ s \\ \quad\quad \textbf{else}\ reduce(f, f(s, head\ ts), tail\ ts) \end{array}$$

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

Sets may be used to represent sequences with no duplicates. Unlike a sequence, a set is not enumerable; but any conceivable representation of a set is. This demands the ability to convert a set into a sequence, for operations where a sequence is more suitable. The following definitions provide support for such an operation.

*take*1 takes an arbitrary element out of a set and returns a pair consisting of the element and the remaining set.

$$[T]$$

$$take1 : \mathbb{P}_1\ T \rightarrowtail T \times \mathbb{P}\ T$$

$$\forall\, ts : \mathbb{P}_1\ T \bullet \exists\, t : ts \bullet take1(ts) = (t, ts \setminus \{t\})$$

takeAll iteratively applies *take*1 until the set is empty, and constructs a sequence from the resulting elements.

$$[T]$$

$$takeAll : \mathbb{P}\ T \rightarrow \operatorname{seq} T$$

$$\forall\, ts, s : \mathbb{P}\ T \bullet \forall\, t : T \bullet (t, s) = take1(ts) \Rightarrow takeAll(ts) = \langle t \rangle \frown takeAll(s)$$

$$[S, T]$$

$$reduce' : (S \times T \rightarrow S) \times S \times \mathbb{P}\ T \rightarrow S$$

$$\forall\, f : S \times T \rightarrow S \bullet \forall\, b : S \bullet \forall\, as, s : \mathbb{P}\ T \bullet \forall\, a : T \bullet (a, s) = take1(as) \Rightarrow reduce'(f, b, as) = \textbf{if}\ \#\, as = 0\ \textbf{then}\ b\ \textbf{else}\ reduce'(f, f(b, a), s)$$

$$sum : \operatorname{seq} \mathbb{N} \rightarrow \mathbb{N}$$

$$\forall\, plus : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N} \bullet \forall\, x, y : \mathbb{N} \bullet \forall\, as : \operatorname{seq} \mathbb{N} \bullet plus(x, y) = x + y \Rightarrow sum(as) = reduce(plus, 0, as)$$

Teller

$$tally : candidates \rightarrow \mathbb{N}$$

$$\forall\, c : candidates \bullet \forall\, v : voters \bullet \forall\, i : \mathbb{N} \bullet \forall\, votes : candidates \rightarrow \operatorname{seq} \mathbb{N} \bullet votes(c)(i) = \textbf{if}\ ballot(v) = c\ \textbf{then}\ 1\ \textbf{else}\ 0 \Rightarrow tally(c) = sum(votes(c))$$