Combinatorial functions

Mapping functional

For an **unary** function f we designate by Map_f the **unary** function taking a list x and **mapping** $f(a)$ to every element of $a \,\varepsilon\, x$, i.e:

$$Map_f(x_1, \ldots, x_n, 0) = f(x_1), \ldots, f(x_n), 0$$

Formally:

$$Map_f(x \oplus (a, y)) = Map_f(x) \oplus (f(a), Map_f(y))$$

Note that $a \in Map_f(x) \leftrightarrow \exists b \,\varepsilon\, x \; a = f(b)$.

We will often use the **dot** notation to construct functions f. For instance $Map_{6+.}$ is Map_f where $f(x) = 6 + x$

Interleave

For a list x and element a we wish to construct the list $Inter(a, x)$, **interleaving** a into the list x at all possible positions. We wish

$$s \oplus (y, t) = Inter(a, x) \leftrightarrow \exists x_1 \exists x_2 (\\ x = x_1 \oplus x_2 \wedge L(x_1) = L(s) \wedge y = x_1 \oplus (a, x_2))$$

This can be achieved by

$$Inter(a, 0) = (a, 0), 0$$
$$Inter(a, b, x) = (a, b, x), Map_{(b,\cdot)} Inter(a, x)$$

Typing notation

We designate by a, b, c, ... **elements** of **lists** x, y, z, ... Combinatorial functions such as $Inter$ yield **lists of lists** which we will designate by s, t, r, ...

We can display such lists by **formats**. For instance, $Ln(x)$ displays the number x as a list of of **decimal** numbers, no matter how x is internally represented. $Str(x)$ displays the list x as a string (list of ascii characters) if possible.

The format Ls displaying a list of strings can be defined as:

$$Ls(0)$$
$$Ls(x,t) \leftarrow Str(x) \wedge Ls(t)$$

Permutations

We call the list x a **permutation** of the list y, in writing $x \sim y$, when

$$x \sim y \leftrightarrow \forall a\, \#(a, x) = \#(a, y)$$

where $\#(a, x)$ **counts** the number of occurrences of a in x. We have

$$x \sim 0 \leftrightarrow x = 0$$
$$x \sim a, y \leftrightarrow \exists x_1 \exists x_2 (x = x_1 \oplus (a, x_2) \wedge x_1 \oplus x_2 \sim y)$$

List of all permutations

We wish to write a function $Perms(x) = t$ where for all y we have $y \varepsilon t$ iff $y \sim x$. We have

$$Perms(0) = 0, 0$$
$$Perms(a, x) = \bigoplus Map_{Inter(a,\cdot)} Perms(x)$$

and $\bigoplus t$ **concatenates** all lists $x \varepsilon t$.

Sorting

We call a list x **increasingly sorted** if $Ord(x)$, i.e. x is **non-decreasing** where

$$Ord(x) \leftrightarrow \forall x_1 \forall x_2 \forall a \forall b (x = x_1 \oplus (a, b, x_2) \rightarrow a \leq b)$$

We call a unary function f a **sort** if

$$\forall x (f(x) \sim x \wedge Ord f(x))$$

Insertion sort

$$Is(0) = 0$$
$$Is(a, x) = Ins(a, Is(x))$$

where

$$Ins(a, 0) = a, 0$$
$$Ins(a, b, x) = b, Ins(a, x) \leftarrow a > b$$
$$Ins(a, b, x) = a, b, x \qquad \leftarrow a \leq b$$

Not a very good sort, because it is $\mathcal{O}L(x)^2$

Merge sort

Merge sort is optimal $\mathcal{O}(L(x)\cdot \log(x))$. Its strategy is **divide et impera**: split the task in two and recur.

$$
\begin{aligned}
Ms(x) &= x && \leftarrow L(x) \leq 1 \\
Ms(x) &= Merge(Ms(y), Ms(z)) && \leftarrow L(x) > 1 \wedge \\
& && \leftarrow Split(x) = y, z
\end{aligned}
$$

Where $Split(x) = y, z$ with $y \oplus z \sim x$ (say, put the elements of x alternatively in the two output lists) and $Merge$ **merges** two nondecreasing lists such that $Merge(x, y) \sim x \oplus y$ and the list is nondecreasing.