Ordered Swaps

by Sven Nilsen, 2020

Order swaps is a list of swapping instructions that generates a permutation.

ordered_swap : [(nat, nat)]

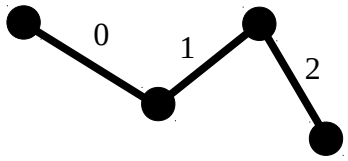
Each swap instruction is stored in the form `(a, b)` where `a < b`.
This is because swapping `a` with `b` is the same as swapping `b` with `a`.
Swapping `a` with itself does not change anything, so `a < b` restricts the swaps to useful ones.

The number of valid swap instructions for a list of size `n`, is equal to the number of pairs of `n`:

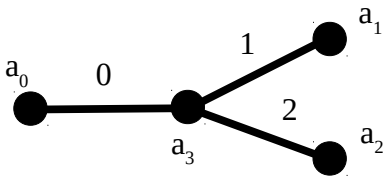
pairs(n) = n · (n – 1) / 2

For example, for a list of 4 elements, the number of pairs is `4 · (4 – 1) / 2 = 6`.

Ordered swaps can be represented visually as edges between nodes labeled with natural numbers:



Ordered swaps can also create a branch structure:



It is helpful to build an intuition for how values get transported along ordered swaps.

If two disjoint sets of edges are labeled `[0, k)` and `[k, m)`,
then they can be relabeled `[m – k, m)` and `[0, m – k)` respectively.

A disjoint set of edges can be dissected and normalized with labels `[0, m)` for some `m`.
A such set without cycles is called a “piece” and represents a local Hamiltonian path.

- All ordered swaps can be constructed from pieces
- All pieces with branches can be normalized to a piece with no branches
- A piece with no branches labeled `[0, m)` has `m + 1` representations
- For every piece on a total order of nodes there exists a canonical piece with no branches