Lecture No. 6, 10/3, Topics

Confluently persistent deques.

Recursive definition of a list as a prefix, a list of pairs, and a suffix.

Use overwriting to create green prefixes, suffixes (good for both push and pop).

Adding catenation (steques):

Objects on recursive list are more complicated: each pair consists of a prefix and a steque of pairs.

Note one important technical point, missed in class: certain cases of pop are implemented naively (without recursion).

This means that only a naïve pop can cause a catenation. See pp. 971-972 of the paper.

The idea generalizes to deques, but seems to require *two* recursive sublists separated by a middle buffer, rather than just one.

Open Problem #6: Obtain confluently persistent deques with catenation and O(1) amortized time bounds using a representation with only one recursive sublist. See p.976 of the paper.

H. Kaplan, C. Okasaki, and R. Tarjan, "Simple confluently persistent catenable lists," *SIAM J. Comput*. 30 (2000), pp. 965-977.