Burrows-Wheeler Transform

substring searching and genetics

Algorithm Basics

Transformation				
Input	All Rotations	Sorting All Rows in Alphabetical Order by their first letters	Taking Last Column	Output Last Column
^BANANA|	^BANANA| |^BANANA A|^BANAN NA|^BANA ANA|^BAN NANA|^BA ANANA|^B BANANA|^	**A**NANA|^B **ANA**|^BAN **A**|^BANAN **B**ANANA|^ **N**ANA|^BA **NA**|^BANA ^BANANA| |^BANANA	ANANA|^**B** ANA|^BA**N** A|^BANA**N** BANANA|^ NANA|^B**A** NA|^BAN**A** ^BANANA| |^BANAN**A**	BNN^AA|A

- From: http://en.wikipedia.org/wiki/Burrows%E2%80%93Wheeler_transform#Example

Interface

- In-place
  - template<typename BidirectionalRangeT>
    void burrows_wheeler(BidirectionalRangeT &);
- Allocating
  - template<typename SequenceT>
    SequenceT burrows_wheeler_copy(const SequenceT &);
- Output iterator
  - template<typename OutputIteratorT, ForwardRangeT>
    OutputIteratorT burrows_wheeler_copy(OutputIteratorT, const ForwardRangeT &);

Interface, cont

- In-place
  - template<typename BidirectionalRangeT>
    void inverse_burrows_wheeler(BidirectionalRangeT &);
- Allocating
  - template<typename SequenceT>
    SequenceT inverse_burrows_wheeler_copy(const SequenceT &);
- Output iterator
  - template<typename OutputIteratorT, ForwardRangeT>
    OutputIteratorT inverse_burrows_wheeler_copy(OutputIteratorT,
    const ForwardRangeT &);

Thoughts

- Works with arbitrary character types?
- No need for locales at this point. May need if arbitrary / lexicographical sorting is wanted
- May add defaulted OutputSequenceT to burrows_wheeler_copy sequence version
- Loose concept requirements for slow implementation
- May have to tighten requirements when adding / changing to optimized implementation