TFA File Merge Algorithm

This document describes the algorithm for merging two TFA (Tabix-indexed TFASTA) files.

Overview

The TFA merge algorithm combines two TFA files by concatenating their DNA sequences while maintaining sample information and sequence structure. The algorithm ensures that both files have compatible sequences and record counts before merging.

Command Line Usage

```
tfa_merge -i file1.tfa.gz -i file2.tfa.gz -o merged.tfa.gz
```

```
Options:
- `-i, --input FILE`: Input file (specify twice for two files)
- `-o, --output FILE`: Output file name
- `-f, --force`: Force overwrite of output file
- `-h, --help`: Show help message
```

Algorithm Steps

```
ALGORITHM MergeTFAFiles(file1, file2, outputFile):

1. VALIDATION PHASE:
   IF file1.numberOfSequences != file2.numberOfSequences THEN
       RETURN error "Different number of sequences"

   // Create sequence mapping
   sequenceMap = new Map()
   FOR EACH seq1 IN file1.sequences:
       found = false
       FOR EACH seq2 IN file2.sequences:
           IF seq1.name == seq2.name THEN
               IF seq1.recordCount != seq2.recordCount THEN
                   RETURN error "Record count mismatch"
               sequenceMap[seq1] = seq2
               found = true
               BREAK
       IF NOT found THEN
           RETURN error "Sequence not found in second file"

2. HEADER WRITING PHASE:
   Write "##fileformat=TFAv2.0" to outputFile
   Write command line info to outputFile
```

```
    // Merge and write sample names
    Write "#NAMES: "
    FOR EACH sample IN file1.samples:
        Write sample.name + " "
    FOR EACH sample IN file2.samples:
        Write sample.name + " "
    Write newline

3. SEQUENCE MERGING PHASE:
   FOR EACH sequence IN file1.sequences:


        iterator1 = createIterator(file1, sequence)
        iterator2 = createIterator(file2, sequence)

        WHILE hasNext(iterator1) AND hasNext(iterator2):
            record1 = next(iterator1)
            record2 = next(iterator2)

            // Verify positions match
            IF record1.position != record2.position THEN
                RETURN error "Position mismatch"

            // Merge DNA sequences
            mergedSequence = record1.sequence + record2.sequence

            // Write merged record
            Write sequence.name + "\t" + record1.position + "\t" +
mergedSequence + "\n"

4. CLEANUP PHASE:
   Close all files
   Free memory
   Create index for output file
```

File Format

Input Files

- Must be in TFAv2.0 format
- Must be tabix-indexed
- Must contain compatible sequences and positions

Output Format

Each line in the output file follows this format: sequence_name position merged_DNA_sequences

Where:

- `sequence_name`: Name of the chromosome or scaffold
- `position`: Position in the sequence (1-based)

- `merged_DNA_sequences`: Concatenated DNA sequences from both input files

Error Conditions

The algorithm will fail if:

1. Input files have different number of scaffolds
2. A scaffold exists in one file but not in the other
3. Matching scaffolds have different record counts
4. Position values don't match during merging
5. Input files are not in TFAv2.0 format
6. Input files are not properly indexed

Implementation Notes

1. Sequence Order
    - Files may have scaffolds in different orders
    - The scaffold mapping handles this difference
    - Output follows the sequence order of the first file
2. Sample Names
    - Sample names from both files are preserved
    - Order is maintained: file1 samples followed by file2 samples
3. Memory Usage
    - Files are processed in streaming fashion
    - Only one record from each file is held in memory at a time
4. Performance
    - Uses tabix indexing for efficient random access
    - Parallel processing of both input files