CS32 Project 4 Report

Junhong Wang 504941113

**Whether any of your classes have known bugs or other problems that we should know about. For example, if you didn’t finish the*GenomeMatcher::findRelatedGenomes()* method or it has bugs, tell us.**

No

**Whether or not each method satisfies our big-O requirements, and if not, what you did instead and what the big-O is for your version.**

Yes, it satisfies all big-O requirements.

**How two of your methods work — use high-level pseuocode to describe them:**

*Trie's find() method*

*If node is null, then return empty vector.*

*Check if the first character matches the first character of key, and get its index (index of corresponding child node)*

*If first character does not match, return empty vector*

*Recursively check rest of children nodes against the rest of key with findHelper()*

*findHelper(Node\* node, const std::string& key, bool exactMatchOnly)*

*If node is null, then return empty vector*

*If key is empty, then return values of node*

*Check if first character matches the first character of key, and get its index (index of corresponding child node)*

*If exactMatchOnly and first character does not match, return empty vector.*

*If exactMatchOnly and first character matches, recursively check rest of children nodes against the rest of key with findHelper()*

*If not exactMatchOnly, for each children nodes, recursively find matches with findHelper() (set exactMatchOnly to true if the index of child node does not match index found in previous step)*

*Return all the matches from all the children nodes*

*GenomeMatcher's findGenomesWithThisDNA() method*

*If length of fragment is less than minimumLength, then return false*

*If minimumLength is less than minSearchLength, then return false*

*Search positions of genome sequences that match first minSearchLength characters of fragment with find()*

Find genome candidates that satisfy constraints from previous search results with *findGenomeWithThisDNACandidates()*

*If candidates are empty, then return false*

find best candidates from candidates with *findBestCandidates()*

*Return true*

*findGenomeWithThisDNACandidates(const string& fragment, int minimumLength, bool exactMatchOnly, vector<GenomePointer>& values, vector<DNAMatch>& candidates)*

*For each value in values (values contains the search results):*

*check how many following characters match the key.*

*If the matched length is greater than minimumLength, then append the match to candidates*

findBestCandidates(*vector<DNAMatch>& candidates, vector<DNAMatch>& matches)*

*Create a dictionary that maps from string to DNAMatch*

*For each candidate in candidates:*

*If candidate is not in the dictionary, add it to dictionary*

*If candidate is already in the dictionary and the new candidate has longer DNA match*

*length, then update the dictionary*

*If candidate is already in the dictionary and the new candidate has same DNA match*

*length and earlier DNA match position, then update the dictionary.*

*For each key, value in the dictionary:*

*Add value into matches*

*Sort matches by DNA match length*