1. System Configuration:

Processor: Intel(R) Core(TM) i7-4700MQ CPU @ 2.40GHz (8 CPUs), ~2.4GHz
Memory: 16384MB RAM

2. Summary of results:

OUTPUT/REPORT for run on synthetic sequences

- Input stats
  - Lamba = 15, 25 proved too restrictive.
  - Reference genome is 5362495 characters
- Main Result
  - Best hits for first 10k synthetic reads is included
- Timing statistics
  - 9596ms to build tree and allocate memory for the program
  - 1083ms to PrepareST()
  - 2023ms to parse the (500k) input reads
  - 889567ms to map 10k sequences and write to outfile
- Other statistics
  - Average alignments per read = 0.4699

OUTPUT/REPORT for run on real read sequences

- Input stats
  - Lamba = 15, 25 proved too restrictive.
  - Reference genome is 5362495 characters
- Main Result
  - Best hits for first 10k reads is included
- Timing statistics
  - 12027ms to build tree and allocate memory for the program
  - 1273ms to PrepareST()
  - 4060ms to parse the (1376k) input reads
  - 836101ms to map 10k reads and write to outfile
- Other statistics
  - Average alignments per read = .07

3. Justification

Run time is pretty abysmal. This is primarily due to the newFindPath() function responsible for finding the starting locations for alignment. Roughly 85% of post construction runtime is spent on this function. Once the problem was narrowed down to that function, improvements have been made but still not enough to parse over a million reads in any reasonable amount of time.

Hit rate is also low. This is because long edges make the “string-depth(u) ≥ x” condition very hard to meet. This could be improved by modifying the algorithm to count all matches rather than only counting matches if they reach a node or the end of the read. This requires at least one more parameter to be passed, and complicates some of the more simple interactions, which is a potential reason for why “string-depth(u) ≥ x” was chosen instead of #matches.

At least time does appear to be linear, taking roughly 80ms per input sequence for 10k, 5k, and 100 inputs. Performance statistics are on the next page in the form of a screenshot, but mapping results are included as txt.

Real sequences:

```
C:\Users\guenw\Documents\gitrepos\471p3\src\471p3\ref>471p3.exe base.fasta DNA_Alphabet.txt real.fasta
The tree took: 12072ms to build including all memory allocation for the program.
The tree took: 1273ms to prepare
It took 4060ms to parse 1376192 reads.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!It t
ook: 836101ms to map 1376192 reads.
Final execution time: 853516 ms.

Reference genome length: 5362496
Reads: 1376192
ms/Read: 0.0126843
Average alignments per read: 0.000564601

Press enter to exit.
```

Synthetic sequences:

```
C:\Users\guenw\Documents\gitrepos\471p3\src\471p3\ref\New folder>471p3.exe base.fasta ..\DNA_Alphabet.tx
t synth.fasta
The tree took: 9596ms to build including all memory allocation for the program.
The tree took: 1083ms to prepare
It took 2032ms to parse 500000 reads.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!It t
ook: 889567ms to map 500000 reads.
Final execution time: 902288 ms.

Reference genome length: 5362496
Reads: 500000
ms/Read: 0.025522
Average alignments per read: 0.009398

Press enter to exit.
```

Note that both alignment/read and ms/read are misreported here, but are correct in the text report above. They assume that all reads were mapped, so I adjusted the value to account for the fact that only 10k reads were mapped. We do see that the synthetic sequences provide many more alignment opportunities, which makes sense considering that fact that they were derived from the reference genome.