Miguel Ricardo Anderson (netId: mra21)

CompSci 201 – Assignment Analysis for DNA

03/24/2016

Part 1

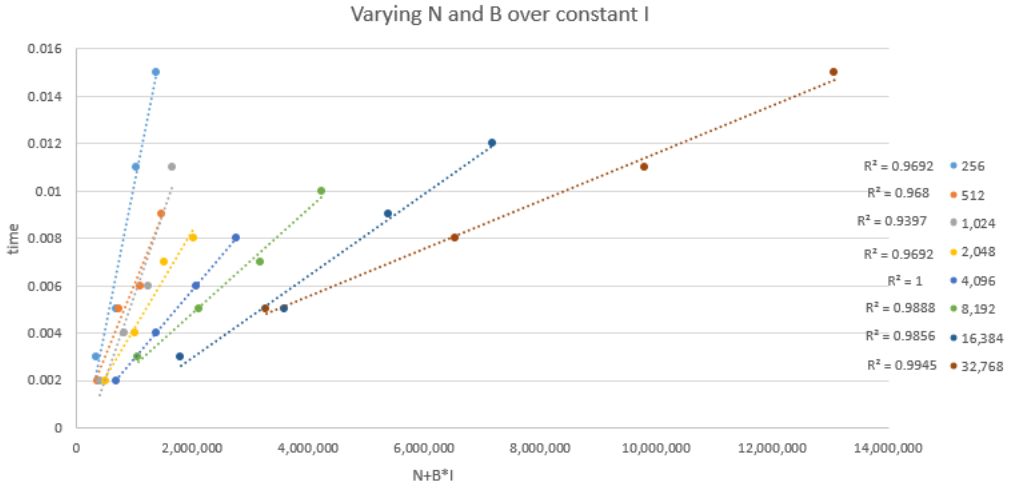
The running time of SimpleStrand's cutAndSplice method is O(n + Bl). Your task is to provide empirical data in the form of graphs, tables, and/or regression equations which demonstrate this relationship. Please also include the process you used to obtain your data, particularly how you solved the n/B relationship.

Value	Description Given	In my words	In code
N	the length of the original strand	Length of text file supplied to DNABenchmark	Dna length
B	the number of occurrences of the enzyme	The number of occurances of the string that is to be replaced. This is close to the value (myAppends) for myAppends is increased every time a splicee is added where an enzyme used to be	~myAppends
I	the splicee length	The length of characters of the string that replaces every iteration of the enzyme	Splicee

Here are some trials that show how time is affected when **N and B are increased and I is held constant**. Each n-th trial is has N'=n*N and B'=n*B with reference to the first trial.

dna length	append times	splicee	recomb	N+B*I	time
320160	90	4,096	504,210	688,800	0.002
640320	180	4,096	1,008,420	1,377,600	0.004
960480	270	4,096	1,512,630	2,066,400	0.006
1280640	360	4,096	2,016,840	2,755,200	0.008

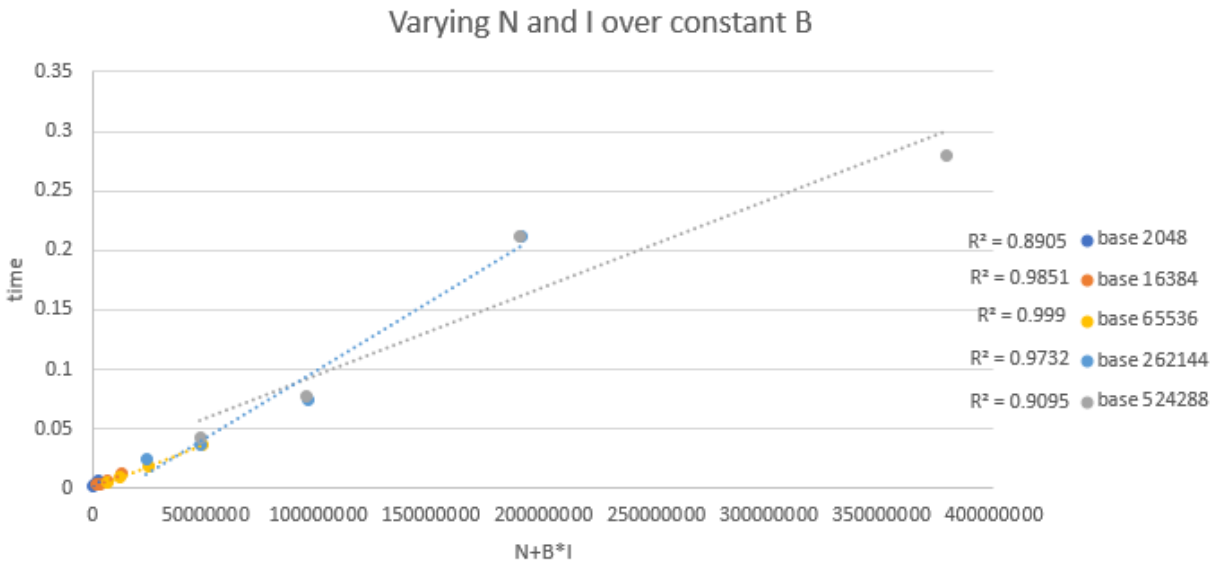
Result: the times follow this pattern and when this is repeated over several different I, or splicee, values we graph the results to see all linear trends. See Part 3 for full summary of SimpleStrand's cutAndSplice



Now let's examine what happens when **N and I are increased and B is held constant**:

dna length	append times	splicee	recomb	N+B*I	time
320160	90	65536	3269010	6218400	0.005
646560	90	131072	6544530	12443040	0.009
972960	90	262144	12769170	24565920	0.019
1299360	90	524288	24892050	48485280	0.036

Result: this also follows the linear trend, see the graph below for a summary of different cases



O(N+B*I) correctly labels the runtime of cutAndSplice in SimpleStrand. The way change N and B individually in the text files is to add the characters "a", "c", "g", and "t" but never in the order of the enzyme. This will ensure the length increases but the occurrences of the enzyme doesn't. To change B without changing N all one has to do is insert the enzyme in place of other parts of the text file (replace not add). This will raise occurrences of the enzyme yet not lengthen the text file.

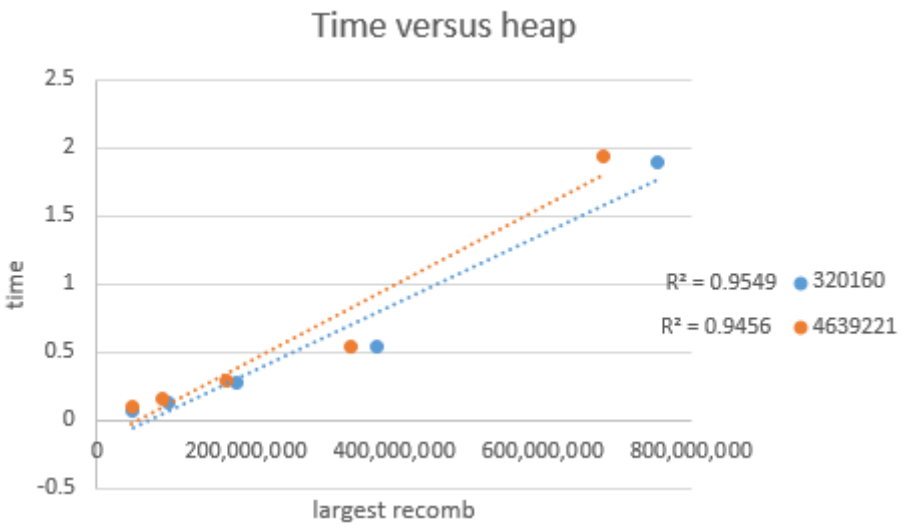
To create these graphs I copy/pasted either text full of the enzyme or text with "a", "c", "g", and "t" but not exactly the enzyme at the end of the text file.

Part 2

This graph summarizes my results (the dna length column denotes whether I used ecoli.txt input file or ecoli_small.txt input file:

dna length	heap	largest recomb	time
320160	-Xmx512M	47,505,810	0.063
320160	-Xmx1024M	94,691,730	0.132
320160	-Xmx2048M	189,063,570	0.275
320160	-Xmx4096M	377,807,250	0.534
320160	-Xmx8192M	755,294,610	1.89
4639221	-Xmx512M	46,906,071	0.101
4639221	-Xmx1024M	89,176,791	0.15
4639221	-Xmx2048M	173,718,231	0.284
4639221	-Xmx4096M	342,801,111	0.533
4639221	-Xmx8192M	680,966,871	1.944

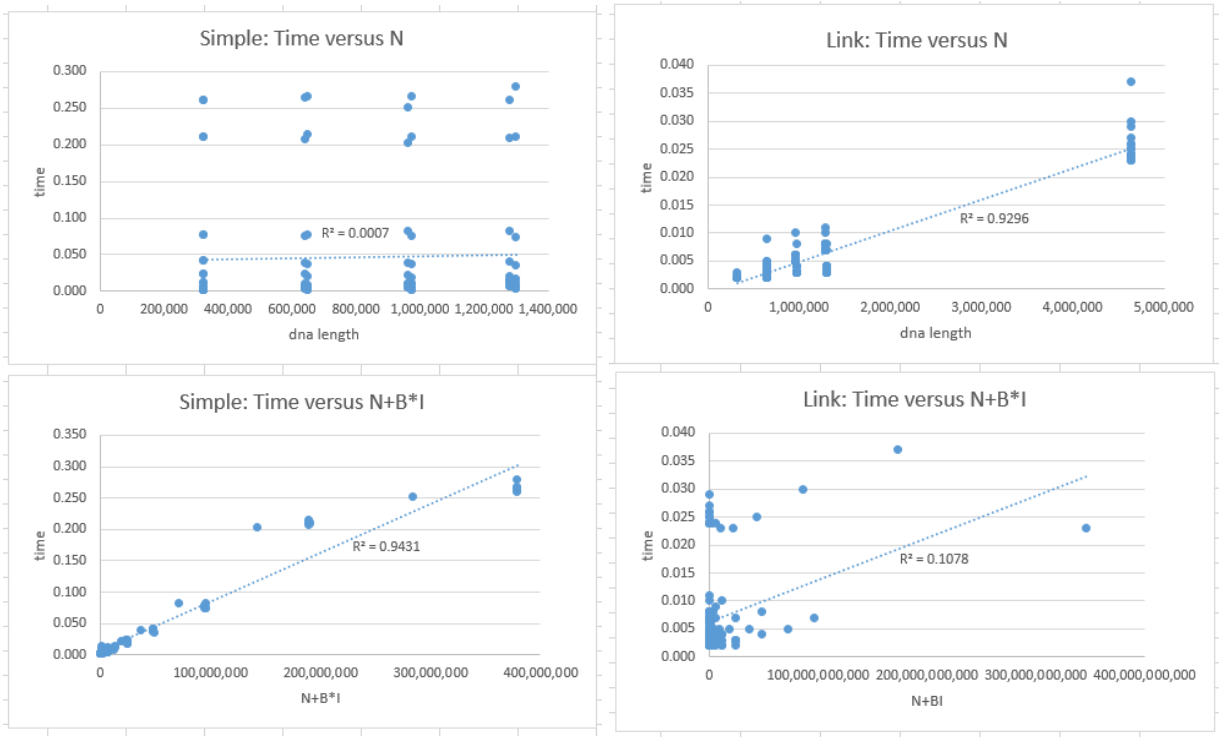
With increases in heap the largest recomb possible increases by nearly factors of 2 each time. They are linearly proportional. These values are also linearly proportional to the time which is shown by this graph.



The runtime is therefore O(n) with n being the largest string size under the circumstances.

Part 3

Benchmark your LinkStrand's cutAndSplice implementation.



These graphs summarize the cutAndSplice runtime: SimpleStrand is O(N+B*I) while LinkStrand is O(N). My hypothesis for the reason LinkStrand does not have a runtime dependence on B or I is because the act of splicing is O(1) in a LinkedList in java. The culprit of this is the .append method. In SimpleStrand this requires much more time than for in LinkedList all you have to do is redirect the last node but in SimpleStrand a StringBuilder and new String is needed because Strings are immutable.