Homework 2

Shaun Schreiber

April 22, 2014

1 Quadratic

1.1 Regular Expressions

1. .+a*c
2. b*b*
3. .+.+F

1.2 Input

1. a. . . aX
2. b. . . bX
3. c. . . cX

1.3 Reason

The .+and symbol* in the above expressions are the culprits that causes the quadratic running time. The reason for it is that it starts by excepting one random character. The number of random characters allowed grows by one each time there is a miss match and because this miss match is at the end it cause quadratic running time. Basically the algorithm backtracks till it finds a branch it hasn't discovered because the branch will only ever be where the number of random characters ends and a new symbol starts it forces the algorithm to run in quadratic time.

2 Cubic

2.1 Regular Expressions

1. .+[cb]*[c]*F
2. a*a*a*F
3. .+.+.+F

2.2 Input

1. c...caX
2. a...aX
3. a...aX

2.3 Reason

To get cubic or any higher exponent we just make use of the following form where n > 0, s_i is distinct and n + 1 is the running time.

$.+[\mathbf{s}_1]^* \ [\mathbf{s}_1\mathbf{s}_2]^* \ldots [\mathbf{s}_1 \ldots \mathbf{s}_n]^*\mathrm{F}$

or

$\underbrace{\mathbf{s}_1}_{0}{}^* \underbrace{\mathbf{s}_1}_{1}{}^* \ldots \underbrace{\mathbf{s}_1}_{n}{}^*\mathrm{F}$

Note that the input string must be created in such a why that it will always only fail at the very last character. The reason why these expressions have cubic running time is because each time a miss match accrues the algorithm backtracks till it finds a branch it hasn't yet tried. In this case these branches are when the first and second pattern meet and when the second and third pattern meet. When all of the branches between the second and third pattern has been tried, only then will it go on to the next available branch between the first and second pattern and then retry all of the branches between the second and third pattern again. Repeating this process till all branches are discovered. This causes the cubic running time.

3 Exponential

3.1 Regular Expressions

1. ((a)*)*
2. ((ab)*)*
3. (a|aa)*

3.2 Input

1. a...aX
2. ab...abX
3. a...aX

3.3 Reason

The reason each of these regular expressions take exponential time is: the inner star indicates that there can be any number of a's the outer star indicates that there can be any number of these groups of a's. Therefore if the very last element forces a miss match it will try all possible group sizes and combinations of these group sizes thus taking exponential time.