Systems Programming Laboratory, Spring 2022

Text processing utilities

Abhijit Das Arobinda Gupta

Department of Computer Science and Engineering Indian Institute of Technology Kharagpur

February 15, 2022

The roadmap

- Unix provides a lot of utilities to process text files (or stdin inputs).
 - **sort** sorts a file line by line.
 - **uniq** removes duplicate lines in a (sorted) file.
 - wc reports the counts of characters, words, and lines in one or more file(s).
 - ...
- Here, we will focus on a few of more sophisticated text-processing tools.
 - Pattern matching based on regular expressions is often very useful.
 - grep is a utility to do a lot of tasks on lines matching patterns.
 - **sed** selects lines in a file based upon line numbers or patterns, and can do a set of simple tasks on the selected lines.
 - **awk** is a full-fledged programming language targeted to handle text databases.

Regular Expressions

Systems Programming Laboratory, Spring 2022 Abhijit Das and Arobinda Gupta

- Same as those introduced in connection with regular languages in your FLAT course.
- Constructs are different.
- Used by less, grep, sed, awk, shells, and many text editors like vi and emacs.
- We use less to demonstrate the matches.
- Running with the –N option lets less show the line numbers.
- In the viewing mode, you can type / (forward slash) followed by a regular expression.
- All matches found are highlighted.
- Searches are made in each line.
- The newline character is not allowed in regular expressions.

Viewing matches with less

1 Abstract

3 This tutorial focuses on algorithms for factoring large composite integers 4 and for computing discrete logarithms in large finite fields. In order to 5 make the expections def-sufficient, 1 start with some common and popular 5 public days algorithms for emergation, Bay advander, and elaytel kinemizers and the start of the start of the start of the start of the start main of the start start of the sublic days protocols.

11 The exponential-time integer-factoring algorithms are first covered: 12 trial division and bollard's none method. This is followed by two 13 sub-exponential algorithms based upon Fernet's factoring method. Union's 15 method uses includent awardras but illustrates the basic covers of the 16 method uses includent awardras but illustrates the basic covers of the 16 Quadrasic Steve method (QSD) which brings the benefits of using shall 17 candidates for smoothness testing and of selving.

¹⁹ As the third module, I formally define the discrete-logarithm problem (DLP) 20 and its variants. As a representative of the square-root methods for solving 21 the DLP, the baby-step-ginal-step nethod is explained. Next, I introduce the 22 index calculus method (ICM) as a general paradigm for solving the DLP. 23 Various states of the basic ICM are exaliand both for origin fields and

24 for extension fields of characteristic two.

1 Abstract

3 This tutorial focuses on algorithms for factoring large composite integers and for computing discrete logarithms in large finite fields. In order to 5 make the exposition self-sufficient, I start with some common and popular o public key algorithms for encryption, key exchange, and digital signatures. 7 These algorithms highlight the roles played by the apparent difficulty of a solving the factorion and discrete-logarithm archime and discrete information and and and and a solven the factorion and discrete-logarithm archime and the fact deciming.

- 9 public-key pro
- 10

11 Two exponential-time integer-factoring algorithms are first covered: 12 trial division and Pollard's rho method. This is followed by two 13 sub-exponential algorithms based upon Fernat's factoring method. Dixon's 14 method uses random squares, but illustrates the basic concepts of the 15 relation-collection and the linear laberbs atopace. Next. J introduce the

- 16 Ouadratic Sieve Method (OSM) which brings the benefits of using small
- 17 candidates for smoothness testing and of sieving.
- 18

¹⁹ As the third module, I formally define the discrete-logarithm problem (DLP) 20 and its variants. As a representative of the square-root methods for solving 21 the DLP, the baby-step-giant-step method is explained. Next, I introduce the 22 index calculus method (ICM) as a general paradian for solving the DLP.

- 23 Various stages of the basic ICM are explained both for prime fields and
- 24 for extension fields of characterist

1 Abstract

This tutorial focuses on algorithms for factoring large composite integers and for computing discrete longerithms in large frints fields. In order to 5 make the exposition self-sufficient, I start with some common and popular 6 public key supporting for energy thin, bey exchange, and digital signatures 7 these algorithms for energy hings by the apparent difficulty of 9 public-key portocols.

10

²¹ Too exponential-time integer-factoring algosithms are first covered: 22 trial division on Pollard's how that this is followed by 33 sub-propertial algosithms based upon Ferret's factoring method, blow's 50 relations officients and the thread algobas subgets. Next, 12 triverodes the 50 relations officients and the thread algobas subgets. Next, 12 triverodes the 50 quadratt Sieve Method (QSM) which brings the basefits of using small 22 candidates for subgets are straining and sieving.

¹⁹ As the third module, 1 formally define the discrete-logarith grablem (DP) 20 and its variants. As a representative of the source roots methods for solving 21 the DP, the baby step-giant-step method is scalained. Next, 1 introduce the 22 index calculos method (TCM) as a general parading for solving the DP. 23 Various stages of the basic ICM are explained both for prime fields and 24 for extension fields of characteristic two.

Systems Programming Laboratory, Spring 2022

Abhijit Das and Arobinda Gupta

Matching any character

- Period (.) matches any single character.
- The pattern **a**.**g** matches the following.

1 Abstract

This tutorial focuses on Experithes for factoring large composite integers and for computing discrete logarithms in large finite fields. In order to 5 make the exposition self-sufficient, I start with some common and popular 6 public key Experithms for encryption, key exchange, and digital signatures. 7 These Experitoms highlight the roles played by the apparent difficulty of 8 activing the factoring and discrete-logarithm problems, for designing 9 public-key protocols.

If the exponential-time integer-factoring Deprintme are first covered: Striel division and Pollard's che nethod. This is followed by two Standburg and the standburg of the strelation-collection and the linear-states the basic concepts of the strelation-collection and the linear-states are basic states. Next, I introduce the strelation collection and the linear states herefits of using small strendburg of strendburg of steving.

39 As the third nodule, I formally define the discrete-logarithm problem (DLP) 20 and its variants. As a representative of the square-root methods for solving 21 the DLP, the baby-step-giant-step nethod is explained. Next, I introduce the 22 index calculus method (10%) as Defense I paradigm for solving the DLP. 23 Various stages of the basic ICM are explained both for prime fields and 24 for extension fields of characteristic two.

Matching a set of characters

- Delimit the set between [and].
- [Tt] matches upper- or lower-case T.
- **[AEIOU]** matches any upper-case vowel.
- [a-z] matches any lower-case letter.
- [a-zA-Z0-9] matches any alphanumeric character.
- The regular expression **[A-Z] [a-z] [a-z]**. gives the following result.

1 Abstract



Negation of a set of characters

- Use ^ after [.
- [^] matches any non-space character.
- [^aeiouAEIOU] matches any character other than the vowels.
- [^a-zA-Z] matches any non-alphabetic character.
- The output for the search [^AEIOU] [^a-zA-Z].....[a-drt] is given below.

1 Abstract 4 and for computing discrete logarithms in large finite fields. In order to 5 make the exposition self-sufficient. I start with some common and popular 6 public-key algorithms for encryption, key exchange, and digital signatures 7 These algorithms highlight the roles played by the apparent difficulty of 8 solving the factoring and discrete logarithm problems, for designing 9 public-key protocols. II Two exponential-time integer-factoring algorithms are first covered: 12 trial division and Pollard's rho method. This is followed by two 13 sub-exponential algorithms based upon Fermat's factoring method. Dixon's 14 method uses random squares, but illustrates the basic concepts of the 15 relation-collection and the linear-algebra stages. Next, I introduce the 16 Quadratic Sieve Method (OSM) which brings the benefits of using small 17 candidates for smoothness testing and of sieving. 19 As the third module. I formally define the discrete logarithm problem (DLP) 20 and its variants. As a representative of the square-root methods for solving 21 the DLP, the baby-step-giant-step method is explained. Next, I introduce the 22 index calculus method (ICM) as a general paradigm for solving the DLP. 23 Various stages of the basic ICM are explained both for prime fields and

Matching zero or more characters

- Use *****.
- . * matches any string. . . * matches any non-empty string.
- [a-z] * matches any sequence of lower-case letters.
- [^a-zA-Z] * matches any sequence of non-alphabetic characters.
- Result of searching [A-Z] [a-zA-Z] * [^] is given below.
- Longest possible matches are reported, starting as early as possible.



Match at the beginning or at the end of a string

- If you want the match to start from the beginning, use ^ as the first symbol.
- If you want the match to finish at the end, use **\$** as the last symbol.
- The pattern ^ [A-Z] [a-z] * matches the first word of a line if the line starts with a capital letter.
- The pattern [a-z] *\$ matches the last word of a line if the line ends with a lower-case letter, and if the last word consists of lower-case letters only.
- The result for searching **^ [A-Za-z**,] ***\$** is given below.



Quoting the special characters

- Use $\, \[, \], *, \^, \$, $\$, and $\/$. The last one is used during substitution.
- - need not be quoted.
- [a-z] * \. matches the last word with the period in a sentence if the word consists of lower-case letters only. If the last word contains characters other than the lower-case letters, then the match starts after the last such character.
- The pattern [a-z] *-[a-z-] *.*\. matches as follows.

