CS3423/CS6240 Lab Exam

Aug 19 Semester

November 12th, 2019

Time: 4 hours Total: 100 marks

Instructions

- You are allowed to access flex/bison manuals ONLY through man pages that are available locally (use **info (man) flex/bison** command).
- The Internet is allowed ONLY to submit your work at the end of the exam. You should inform TAs while submission. Usage of the internet during the course of the exam is **strictly** prohibited. It is assumed that you have read the "**Rules and Logistics**" Document and are familiar with the Plagiarism policy of CSE@IITH.
- Any electronic device apart from laptops that you use for taking this exam is prohibited.
- Any methods to access solutions through unfair means would result in immediate granting of an **FR grade** in the entire course (CS3423 / CS6240).
- You can specify any additional assumptions your implementation makes and/or status (if incomplete) in the **README** file bundled along with the solution.
    - Do note that if you are submitting only a partial solution, your chances of getting partial marks increases if you document your code and give a good README.
- Place all your code files and your readme file in the provided directory structure. *Use submit.sh (bash submit.sh <roll_no>) script to generate a compressed file that you can upload in the Google Classroom.* (Eg: bash submit.sh CS17BTECH00000).
- You can assume that the input is error-free. Meaning, there is no need to do error handling.

Q1. Iptables Rules **(Difficulty: 2.5*; 30 marks)**

Iptables is a Linux utility administration tool for IPv4/6 packet filtering and NAT. **Iptables** is used to set up, maintain, and inspect the tables of IP packet filter rules in the Linux kernel. *(Excerpt: Linux man page)*

In simpler words, the Iptables specifies the action to be taken on an *incoming/outgoing* packet from a specific source IP and the destination IP.

The motive of this question is to use Lex/Yacc to parse the incoming Iptables rules, process it and print the structure of the resulting Iptables. To make the problem easier, let us restrict the actual syntax/semantics of Iptables to consider the following *reduced man page specification* with the restricted *usage, commands,* and *options.*

There can be only *three* chains/tables (all are empty by default): INPUT, FORWARD, OUTPUT

Usage:

iptables -I chain [rulenum] [options]
iptables -R chain rulenum [options]
iptables -D chain rulenum
iptables -P chain target

(Note: Every token in the input is single space-separated.)

Commands:

-D chain rulenum	Delete rule rulenum (1 = first position) from chain
-I chain [rulenum]	Insert in chain as rulenum (default 1=first position)
-R chain rulenum	Replace rule rulenum (1 = first) in chain
-P chain target	Change policy on chain to target (default “ACCEPT”; other options: “DROP”)

Options:

-p proto	protocol: by name, eg. ‘tcp’, ‘udp' (default “all”)
-s address	source IP address (default “anywhere”)
-d address	destination IP address (default “anywhere”)
-j target	target for rule (defines the action, default “”, other options: ‘DROP’, ‘ACCEPT’)

(You can use man page of iptables, if needed.)

Output format:
The output is the resulting three tables corresponding to INPUT, FORWARD and OUTPUT chains after processing the input rules. Each table contains *four* tab <\t> separated columns corresponding to target, protocol, source, and destination respectively.

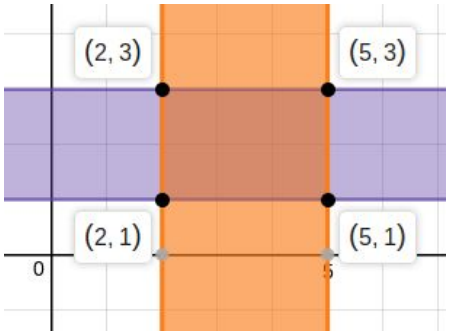
The first line of each table corresponds to the description of the table with the policy. The next line corresponds to the *tab-separated* column headers (4) which are followed by the rules. Every rule starts in a new line. There should be a two-line (\n\n) gap between the two tables.

Sample Input	Sample Output
iptables -I OUTPUT 1 -j DROP iptables -P FORWARD DROP iptables -I INPUT 1 -s 1.1.1.1 -d 2.2.2.2 iptables -D OUTPUT 1 iptables -R INPUT 1 -s 2.2.2.2 -d 1.1.1.1 iptables -I OUTPUT 1 -p udp -j DROP	Chain INPUT (policy ACCEPT) target prot source destination all 2.2.2.2 1.1.1.1 Chain FORWARD (policy DROP) target prot source destination Chain OUTPUT (policy ACCEPT) target prot source destination DROP udp anywhere anywhere

Q2. Polyhedral Calculator **(Difficulty: 4*; 45 marks)**

A system of linear (affine) equations and inequalities can represent a polyhedron. This way of representing a polyhedron with constraints is called the $\mathcal{H}$-form of a polyhedron.

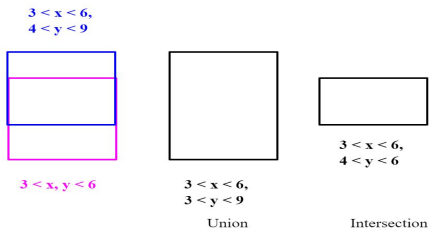
For example: The constraints, $2 \leq x \leq 5$ & $1 \leq y \leq 3$, gives a box polyhedron with (2, 1), (5, 1), (5, 3) and (2, 3) as vertices.



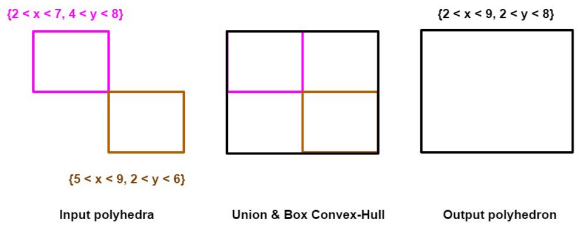
Given a sequence of operations on polyhedra, find the resulting polyhedron and print it. The operations can be *union* or *intersection*. In order to simplify the question, let us assume the input polyhedral constraints will always be two dimensional and is guaranteed to result in a box polyhedron, as shown in the example.

Operations on polyhedra:

Let us consider two operations: Union and Intersection. The example given below shows the resulting polyhedron on performing union and intersection of input polyhedra $\{3 < x, y < 6\}$ and $\{3 < x < 6, 4 < y < 9\}$.



Example: Union & Intersection



Example: The special case of Union

Special case on Union:

Union of two *box polyhedra* is not always a polyhedron. We have to find a convex-hull on the resulting shape, so as to make it a valid polyhedron. In order to simplify this question, let us assume the convex hull to be the *tightest* box that encloses the union of two polyhedra (as shown in the above example).

Assumptions on Input polyhedra and the output polyhedron:

- Polyhedra would always be a *Box polyhedra*.

- Polyhedra would always be in the *Positive Integer domain*, and *strictly greater than zer*o.

Input Format:

First line contains the length of the x-axis (x) and y-axis (y) separated by space. x and y are guaranteed to fit the polyhedron. Second line specifies *n* polyhedra and *n-1* operations, in the following format.

x y

{poly$_1$}op$_1${poly$_2$}op$_2$.....op$_{n-1}${poly$_n$}

- A polyhedron (poly) can be represented in two ways:
  - p<space><ie><space>x,<space>y<space><ie><space>q (Eg: 3 < x, y < 6) and
  - p<space><ie><space>x<space><ie><space>q,<space>r<space><ie><space>y<space><ie><space>s (Eg: 3 < x < 6, 4 < y < 9) ({p, q, r, s} ∈ $\mathbb{Z}^+$)
- Inequalities (ie) of the polyhedron can be < or <=
- Operations (op) can be *u* or *n* for union or intersection, respectively

Output Format:

The output has the plots of *n-1* polyhedra on performing *n-1* operations from *left to right*, in the following format. Points in the polyhedron are represented by 'x'. (As n=2, the output is a single polyhedron. Resulting polyhedron corresponding to the above example: *2 <= x <= 7, 3 <= y <= 6.*)

Sample Input	Sample Output

10 6
{*2 <= x <= 7, 3 <= y <= 6*}u{*3 <= x <= 5, 3 < y < 6*}n{1 <= x,y < 3}
	6| x x x x x x
	5| x x x x x x
	4| x x x x x x
	3| x x x x x x
	2|
	1|
	0|_ _ _ _ _ _ _ _ _ _ _
	  0 1 2 3 4 5 6 7 8 9 10
	
	6|
	5|
	4|
	3|
	2|
	1|
	0|_ _ _ _ _ _ _ _ _ _ _
	  0 1 2 3 4 5 6 7 8 9 10

Q3. Rock, Paper, Scissors! 👊🖐✌ **(Difficulty: 2*; 25 marks)**

At Elan 2020, a competition of Rock (R), Paper (P), Scissors (S) will be held with some enhancements. Your job is to find the winner among N players from the sequence of K moves by each player.

The input specifies K moves for each of the N players followed by the player's *match sequence* (P_1, P_2,..., P_N —> P_1 competes with P_2, whose winner competes with P_3 and so on.)

Rules

1. At a time two players can compete, and both will make unbiased moves.
2. R (👊) beats S(✌); P(🖐) beats R(👊); S(✌) beats P(🖐).
3. Every match is a *knock-out* in nature, where the winner is selected based on the first *non-tie* move. The winner will compete with the next player in the *match sequence*.
4. Tie-breaker: If both players make the same move, then it is a tie. In this case, both players are allowed to make the next move from their sequence of K moves. If the tie cannot be resolved even after K moves, then the player with the most wins across the matches is selected for the next match. If both players have zero wins (first match), the player who is first in the *match sequence* will be declared as winner (FCFS).
5. The sequence of K moves is present in a ***circular queue***. That is, the first move of the winner from the previous match is the next move from his sequence of moves in a *round-robin* fashion.
6. The winner of the last match is the winner of the competition.

Input Format

- The first line of the input specifies N and K (N==<#Players> K==<#Moves>)
- Next N lines describe player id and his moves in a new line.
    - Each Player is given a unique Id (playerID) P<reg_num>. reg_num can be any valid integer.
    - P<reg_num>==<K moves ∈ {R, P, S}>
- Last line specifies the permuted *match sequence* in which players will participate in the competition.

Output Format

- The output should contain the sequence of moves for N-1 matches separated by a line.

- For each match, the first line specifies the participants of that match. <playerID><space>X<space><playerID>
- The next few lines of the match specify the moves and the outcome.
    - <playerID>(<move>)<space>X<space><playerID>(<move>)--><outcome>
    - move ∈ {R, P, S}; the outcome can be T (tie) or the <playerID> corresponding to the winner.
- Last line corresponding to the match declares the winner.
    - <playerID><space>X<space><playerID>--><playerID>
- After the last match, declare the winner of the competition
    - <playerID> is the winner of Rock, Paper, Scissors Competition held at Elan 2020, IITH

Sample Input	Sample Output	Explanation
	P10 X P20 P10(P) X P20(P)-->T P10(S) X P20(P)-->P10 P10 X P20-->P10	Match between P10 and P20 Tie 2nd move of P10 & P20; P10 wins Match won by P10
N==4 K==5 P10==PSRPP P20==PPRPS P30==RPPPS P40==SSSSS P10P20P30P40	P10 X P30 P10(R) X P30(R)-->T P10(P) X P30(P)-->T P10(P) X P30(P)-->T P10(P) X P30(P)-->T P10(S) X P30(S)-->T P10 X P30-->P10 P10 X P40 P10(R) X P40(S)-->P10 P10 X P40-->P10 P10 is the winner of Rock, Paper, Scissors Competition held at Elan 2020, IITH.	Match between P10 and P30 3rd move of P10 & 1st move of P30; Tie 4th move of P10 & 2nd move of P30; Tie 5th move of P10 & 3rd move of P30; Tie 1st move of P10(circular queue) & 4th move of P30; Tie 2nd move of P10 & 5th move of P30; Tie Match won by P10 based on tie-breaker Match between P10 and P40 3rd move of P10 & 1st move of P40; P10 wins Match won by P10 P10 declared as the winner of the game, as he is the unbeaten throughout.

ENJOY THE GAME!

ALL THE BEST
IITH-Compilers-Admin team
