SIGNAL FLOW GRAPH & ROUTH STABILITY CRITIRION

Team Members (Team 7)	**Date: 20/04/2024**
▪ محمد شريف فتحى يوسف جليله	ID: 21011151
▪ آسر أسامه السيد حسن السيد الزكى	ID: 21010241
▪ محمد طارق حسين حسن ابراهيم	ID: 21011161
▪ محمد محمد محمد عبد المنعم الدسوقى	ID: 21011213
▪ ابراهيم محمد السيد مصطفى عمر	ID: 21010023
▪ رنيم احمد السيد شحاته	ID: 21010531

1. Problem Statement

A signal flow graph is a graphical representation used to analyze and describe the flow of signals through a system, and is useful for understanding complex systems with multiple inputs, outputs, and feedback loops.

The Routh-Hurwitz stability criterion is a mathematical method used to determine the stability of a system based on the coefficients of its characteristic polynomial, $1 + GH$, providing a systematic method to analyse the stability of linear systems without directly computing the roots of the characteristic polynomial.

It is required to create the signal flow graph representation of a system, given that total number of nodes and numeric branches gains are given. The program should be able to draw the graph, list all forward paths, individual loops, all combination of n non-touching loops, and give the values of all forward paths' determinants, and the overall system transfer function. The program should also be able to take an input characteristic equation and state whether the system is stable, and if not, the program should list the number and values of poles in the right-hand-side of the s-plane

2. Main Program Features

- Signal Flow Graph graphical representation
- Displaying Detailed Breakdown of Mason's Rule
  - > Forward Paths
  - > Loops and Non-Touching Loops
  - > Determinant of each forward path
  - > Path Gains and Loop Gains
  - > Overall System Transfer Function
- Given a numeric system (path gains are numbers rather than strings), the program calculates the final values of the gains.
- Determining stability of a system using Routh-Hurwitz stability criterion
- Stating values of poles in right-hand-side of s-plane in case of an unstable system

3. Data Structures

(a) Signal Flow Graph Representation

ARRAYS

- `graph` → two-dimensional adjacency matrix representation for the signal flow graph
- `expn` → acts as a container for the two parameters of the method `ExpMult`, `exp1` and `exp2`.
- `paths` → holds all the possible forward paths
- `loops` → holds all loops in the graph
- `paths_loops` → used to find the determinant for each forward path
- `nonTouchingLoops` → holds the non-touching loops
- `uniqueSets` → used to track unique sets based on a serialized key

(b) Routh-Hurwitz Stability Criterion

ARRAYS

- `table` → two-dimensional matrix representing the routh-hurwitz table

4. Main Modules

(a) Signal Flow Graph Representation

- `convertExpression()` → modifies a given expression to work with
- `ExpMult()` → implements exponent multiplication
- `ExpAdd()` → implements exponent addition
- `areNonTouching()` → checks if two loops are non-touching
- `isNumber()` → checks if value is finite and is a number (not NaN)
- `findAllSimplePaths()` → returns a list of all possible forward paths in graph
- `findLoopsFromNode()` → returns a list of all loops in graph
- `paths_loops()` → used to calculate the determinant of a forward path by finding any remaining loop after removal of forward path
- `calculatePathGain()` → returns the path gain of a given forward path
- `calculateLoopGain()` → returns the loop gain of a given loop
- `calculateDelta()` → calculates the system determinant (characteristic function)
- `findNonTouchingLoopsSets()` → returns a list of the non-touching loops in the graph
- `solve()` → the main function used to call the functions and output the results

(b) Routh-Hurwitz Stability Criterion

- `routhInitialTable()` → initializes and returns a two-dimensional matrix representing the routh-hurwitz table using the list of coefficients
- `getElement()` → calculates each table entry using
- `checkZeroRow()` → checks if a given row is a zero row (contains all zeros)
- `zeroRow()` → calculates derivative of row above zero row & sets values of row after zero row
- `routhFinalTable()` → returns the finalized two-dimensional routh-hurwitz table
- `routhStability()` → checks stability of system by checking first column for sign changes
- `getRoots()` → used to solve the function
- `getRootsArray()` → sets the function in an array form to work with it
- `parse()` → returns a list of the coefficients in a given characteristic equation by parsing it
- `routh()` → returns all system data (stability, degree, coefficients, and right-hand-side poles)

5. Algorithms Used

(a) Signal Flow Graph Representation

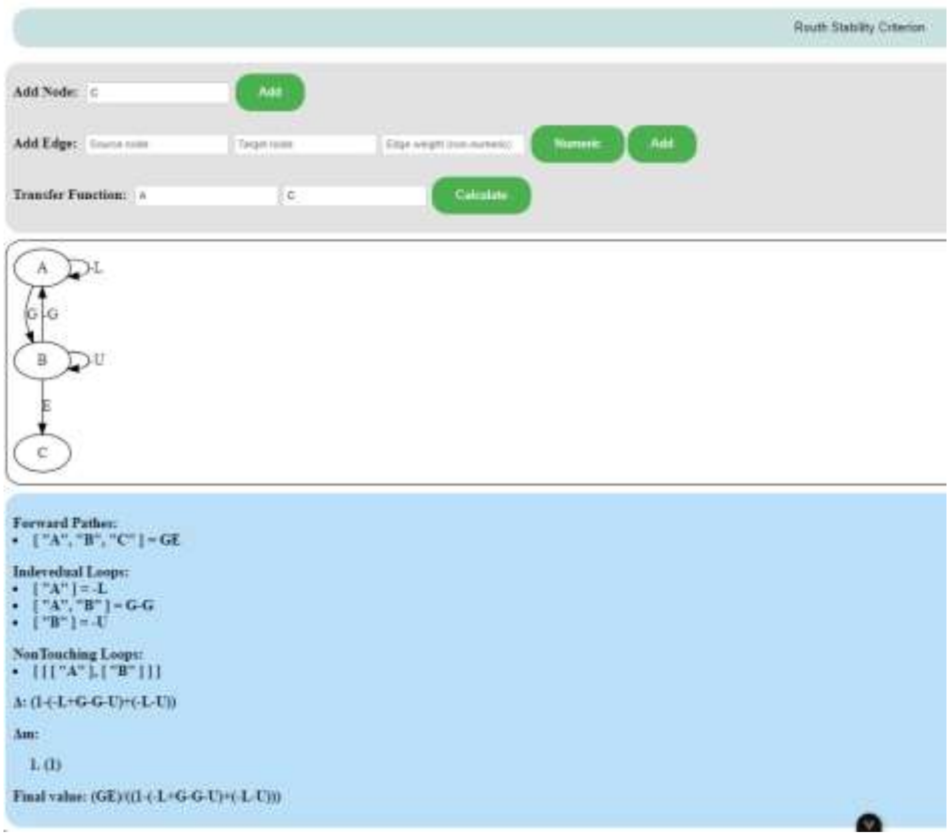
Depth-First Search → used to traverse the graph to find the loops and the forward paths

(b) Routh-Hurwitz Stability Criterion

Routh-Hurwitz Table Method → to calculate each table entry and determine system stability

6. Sample Runs

(a) Signal Flow Graph Representation



(b) Routh-Hurwitz Stability Criterion

Signal Flow Graph

Enter the characteristic equation 1s^3+2s^2+1*s+2 Solve

Result:

System is unstable

Degree of the polynomial: 3

Coefficients: [1, 2, 1, 2]

Routh-Hurwitz Table:

1	1
2	2
0	0
2	0

Number of sign changes in the first column: -1

Stable poles: ["-2", "i", "-i"]

Number of unstable poles: 0

Unstable poles: []

Signal Flow Graph

Enter the characteristic equation s^5+1s^4+10s^3+72s^2+152s+240 Solve

Result:

System is unstable

Degree of the polynomial: 5

Coefficients: [1, 1, 10, 72, 152, 240]

Routh-Hurwitz Table:

1	10	152
1	72	240
-62	-88	0
70.58064516129032	240	0
122.82266910420475	0	0
240	0	0

Number of sign changes in the first column: 2

Stable poles: ["-3", "-1+1.732i", "-1-1.732i"]

Number of unstable poles: 2

Unstable poles: ["2+4i", "2-4i"]

Signal Flow Graph

Enter the characteristic equation

s^5+1s^4+2s^3+2s^2+1s+1

Solve

Result:

System is unstable

Degree of the polynomial: 5

Coefficients: [1, 1, 2, 2, 1, 1]

Routh-Hurwitz Table:

1	2	1	
1	2	1	
4	4	0	z
1	1	0	
0	0	0	
1	0	0	

Number of sign changes in the first column: -1

Stable poles: ["-1", "i", "-i", "i", "-i"]

Number of unstable poles: 0

Unstable poles: []

Signal Flow Graph

Enter the characteristic equation

s^4+1s^3-3s^2-1s+2

Solve

Result:

System is unstable

Degree of the polynomial: 4

Coefficients: [1, 1, -3, -1, 2]

Routh-Hurwitz Table:

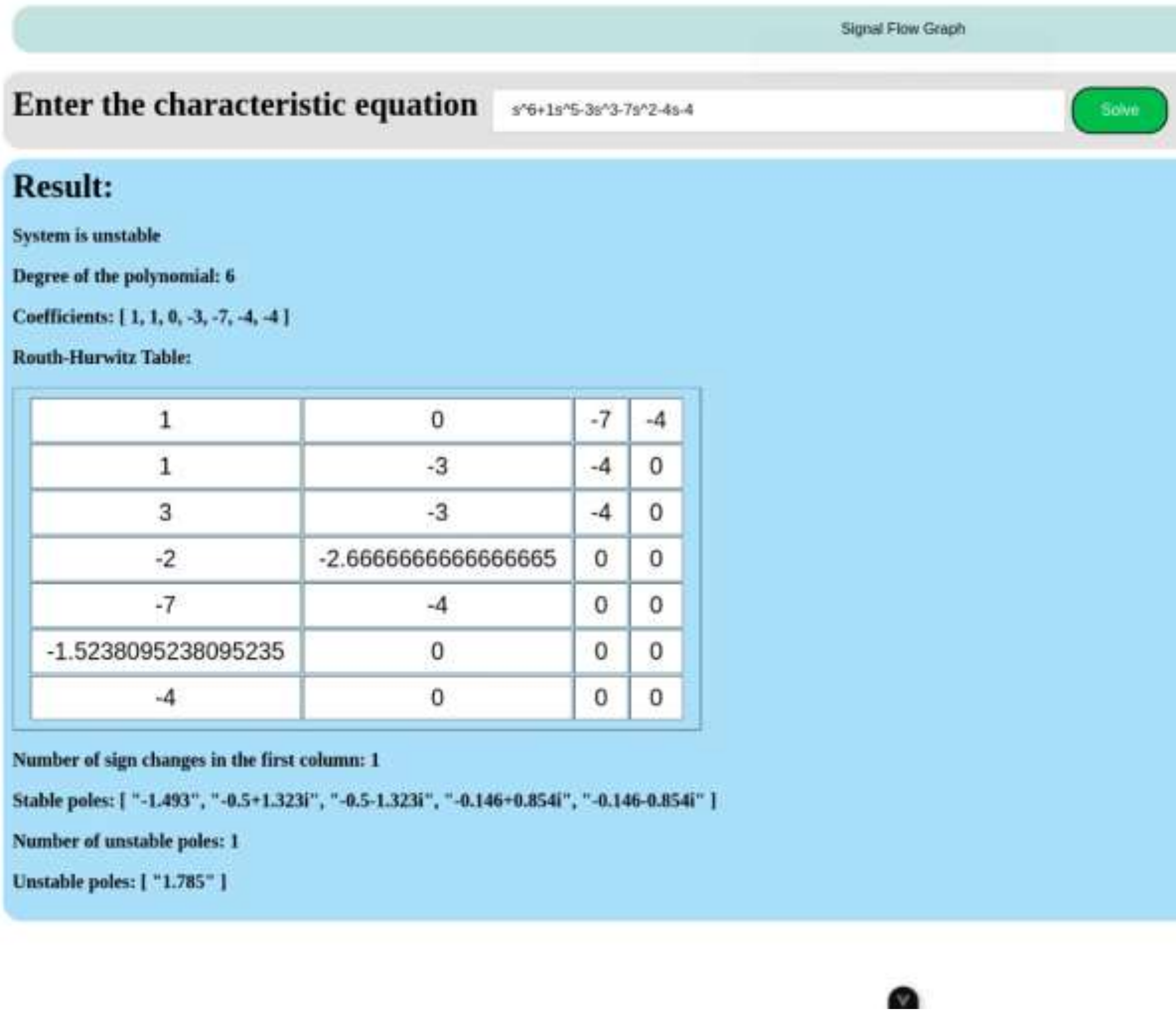
1	-3	2
1	-1	0
-2	2	0
0	0	0
2	0	0

Number of sign changes in the first column: -2

Stable poles: ["-2", "-1"]

Number of unstable poles: 2

Unstable poles: ["1", "1"]



7. User Guide

The user is first met with a screen containing a bar at the top, as well as three fields: Add Node, Add Edge, and Transfer Function.
To add a node, the user types in the identifier of the node (e.g. “A”), then presses “Add”, which would then display a node in the empty space below the fields.
To add an edge between two nodes, the user must specify the source node and the target node using their identifiers, and specify whether the path gain would be numeric or non-numeric through the “Numeric/Non-numeric” button, which switches the field to accept numeric values if “Numeric” is pressed. After specifying the source, the target, and the gain, the user should press “Add”, which would graphically display the edge between the two nodes.
After setting up the graph, to calculate the overall transfer function and display all graph information, the user should specify the start and end nodes in the “Transfer Function” field, then press “Calculate”, which would display list of forward paths and their determinants, individual loops, non-touching loops, characteristic equation, and transfer function.

To switch to the Routh-Hurwitz Stability Criterion calculator, the user should press the “Routh Stability Criterion” text in the bar at the top, which would reroute them to the fields required for calculating the stability.
The user will be met with only one field in which they will write their system’s characteristic equation. The characteristic equation should be written in the form <sign><coefficient>s^<exponent> (e.g. 14s^3+2s^2+9s+1).

After pressing “Solve”, the stability of the system will first be stated, and all the system information (polynomial degree, coefficients, number of unstable poles and their values, number of sign changes) and Routh-Hurwitz table will be displayed.

8. Executables and source codes (git repository):

> https://github.com/asserelzeki6/Control_Systems_Solver.git → master branch