ECE655 Project

Calculate the reliability of a serial/parallel and non-serial/parallel system

WU, BO (998-01-0144)
Email: bwu@ecs.umass.edu

1. Interface of the project

There are 4 independent windows: (1) Input Window, (2) Run Button, (3) Output Window and (4) additional function buttons to calculate the reliability of the system.

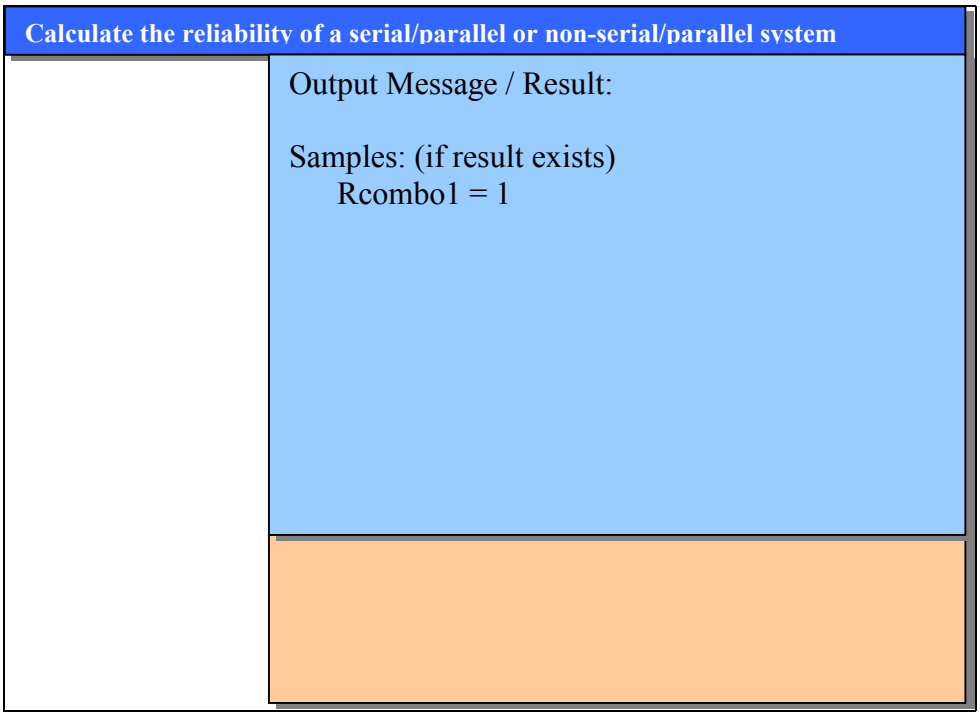

Figure 1. The GUI of the project

2. System Input

Users can input system configuration information from Input Window.
The following parameters should be specified by users:

(1) The number of total nodes in system.
(The node is defined to be the connection point of modules. The index of nodes starts from 1)

(2) The number of total modules in system.

(The index of modules starts from A)

(3) The node pairs between which each module is located. (From input node to output node)

(4) Input node of the entire system.

(5) Output node of the entire system.

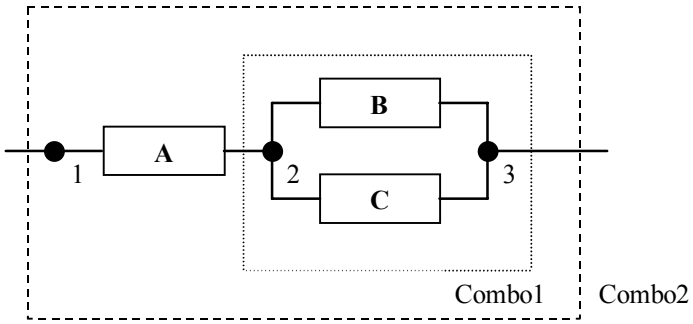

Figure 2. An example of a serial/parallel system

For example, there are totally 3 nodes and 3 modules in this system. The input node of this system is node #1 and the output node is node #3.

The input description of this system is:

Description	*Meaning*
NodesNum = 3	// 3 nodes in this system [1…3]
ModulesNum = 3	// 3 modules in this system [A…C]
A = (1,2)	// A is located between node 1 and node 2
	// and direction is from 1 to 2
B = (2,3)	// B is located between node 2 and node 3
	// and direction is from 2 to 3
C = (2,3)	// C is located between Node 2 and node 3
	// and direction is from 2 to 3
InputNode = 1	// Node 1 is the input of this system
OutputNode = 3	// Node 3 is the output of this system

3. System Output

After input the description of system, the user can click the RUN button to calculate the general expression for the reliability of the system.

(a) If there are some errors in the description, warning/error messages will be given in Output window.

For example, there is an error in line 3 with the following description because node 4 is invalid.

```
NodesNum = 3
ModulesNum = 3
    A = (1,4)
    B = (2,3)
```

C = (2,3)
InputNode = 1
OutputNode = 3

(b) If all descriptions are right, then the result will be printed in Output window.

For the above example, the result will be:
Rcombo1 = (1 – (1– (Rb))*(1–(Rc)))
Rsystem = (Ra)*(Rcombo1)

The first line means that Reliability of the combo system consisting of Module B and C is (1-(1-Rb)(1-Rc)) because Module B and C are parallel modules.
The second line is the final solution for the system reliability because Module A and Combo_1 are serial.

4. Other function modules for the system

(a) Check-Input-File-Module
Input: Configuration file
Output: Check whether the syntax of the configuration file is correct or not.

(b) Calculate-Reliability-Value-Module
Input: the values of reliability for each module
Output: the value of reliability for the entire system

(c) Calculate-Reliability-Expression-Module
Input: the failure rate of each module
Output: the expression of reliability as a function of failure rates for the entire system

5. Data Structures

(a) NodeStruct

```
Struct NodeStruct {
        Short int nodeTag;      // Node tag = 1,2,...;
        Short int nodeOrder;  // Node order
}
```

Notes:
Each node has different value of nodeTag.
Each module is located between the corresponding node pair {from, to}. One is the input node and the other is the output node. In order to describe the input-output relationship, we need nodeOrder to

construct the flow direction of the entire system topology. For any module, nodeOrder of the input node should be less than that of the output node. The nodeOrder of system input node is 1.

For example:

In Figure 2, the nodeOrder of node 1, node 2, node 3 equal 1, 2, 3 respectively.

In Figure 3, the nodeOrder from node 1 to node 7 will be 1, 2, 4, 6, 7, 3, and 5 respectively. That means:

Node 1 ------ > Order 1
Node 2 ------ > Order 2
Node 6 ------ > Order 3
Node 3 ------ > Order 4
Node 7 ------ > Order 5
Node 4 ------ > Order 6
Node 5 ------ > Order 7

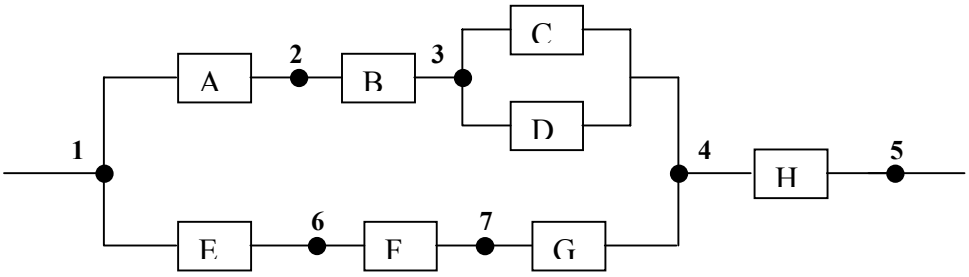

Figure 3. A complex serial/parallel system

(b) ModuleStruct

```
Struct ModuleStruct {
        Struct NodeStruct in;          // input node
        Struct NodeStruct out;         // output node
        String moduleName;             // module name = A, B, ...
                                       // or Combo1, Combo2, ...
        String rel_expression;         // brief expression for reliability
        String rel_full_expression     // full expression for reliability
        Double rel_value;              // value for reliability
        Short int workMode;            // 0 – offline
                                       // 1 – online
        String notes;          // some information for the module
}
```

Notes:

(1) Each Module has its input node and output node.
(2) Each Module has its own module name.
    For example: In Figure 2, Module A's name is "A", module B's name is "B" and module C's name is "C".
(3) The brief expression of reliability for Module A is (Ra).
    The brief expression of reliability for Module B is (Rb).
    The brief expression of reliability for Module C is (Rc).

(4) The rel_value is the numerical value of reliability of the module.

(c) SystemStruct

```
Struct SystemStruct {
        Short int totalNodes;
        Short int totalModules;
        Short int inputNode;
        Short int outputNode;
        Struct ModuleStruct modules[];
}
```

Notes:
We can get all these information from input configure file.

(d) PathinfoStruct

This structure is used store the information of the paths from system in to system out.

```
struct PathinfoStruct
{
    int path[][];            // the paths from system in to system out
    int pathlen[];           // the lengths of the paths
    int  descendorder[]      // The index of paths in descending order
                             //  according to the lengths of the paths
}
```

(e) Structures for expression of the system reliability

1) Struct TermStruct

```
{
 short int  type = 0;    // The type of this term  1: number such as 1,
                         //  2.5.. ,     ; 2: poly like 2*Ra^5*Rc
 int    coeff = 0;       //  if type==1, coeff will be the term's value,
                         // or it will be the coefficient of this term.
 int    mod[];           // the modules' index involved in this term;
 int    modpow[];        // the power of each multiple in this term;
 int    nmod = 0;        // the number of modules involved in this term
}
```

For example, the term $3*R_a^5*R_c*R_d^6$ will be stored as

```
Term{
   type = 2;
   coeff = 3;
   mod[] = { ia,ic,id };
   modpow = {5, 1, 6};
   nmod = 3;
}
```

2) Struct PolynStruct

```
{
  TermStruct   term[];  //  the terms of this expression
  int   nterm[];        // the number of terms in this expression
```

```
    int   ptrstart;          // the pointer to the first term in term[];
    int  ptrend;             // the pointer to the last term in term[]
}
```

6. Algorithms

(a) Algorithm for assigning the values of nodeOrder

It is very easy to assign the values of nodeOrder to guarantee our requirements that for each module the input node's nodeOrder must be less than the output node's nodeOrder.

We can use Topological Sorting algorithm for Directed Acyclic Graph. For this purpose, we represent the nodes as vertices set V, and the node pairs for each module as edges set E, then G = (V, E) is a unweighted directed acyclic graph. So we can easily assign the value of nodeOrder to each node from input to output according to topological sorting.

(b) Algorithm for get the expression of the system reliability

As indicated in notes pp23

$$R_{system} \leq 1 - \Pi \left(1 - R_{pathi}\right) \qquad (7.20)$$

So, the algorithm here find the paths first then iterates on each path to obtain the expression on the right-hand side of (7.20) for this system. Finally derive the exact reliability expression by replacing every occurrence of R_i^j by R_i .

By the algorithm described above, although this program is designed to solve the problem of non-serial/parallel system, it can be used for the serial/parallel system too. This is justified by the included examples on my webpage.