The Loop-breaking Algorithm by Tarjan

- In this lecture, a procedure shall be introduced that is able to break all algebraic loops systematically and algorithmically.
- The ***Tarjan algorithm*** consists of a graphical technique to simultaneously sort systems of equations both horizontally and vertically. The algorithm can furthermore be used to detect algebraically coupled systems of equations.

Table of Contents

The Structure Incidence Matrix I

- The structure incidence matrix contains one row for each equation of the DAE system, as well as one column for every unknown of the equation system.
- Since a complete equation system contains always exactly as many equations as unknowns, the structure incidence matrix is quadratic.
- The element ***<i,j>*** of the structure incidence matrix concerns the equation ***#i*** and the unknown ***#j***. The element assumes a value of ***1***, if the indicated variable is contained in the considered equation, otherwise the corresponding matrix element assumes a value of ***0***.

The Structure Incidence Matrix: An Example

1: $U_0 = f(t)$

2: $i_0 = i_L + i_{R1}$

3: $u_L = U_0$

4: $di_L/dt = u_L / L_1$

5: $v_1 = U_0$

6: $u_{R1} = v_1 - v_2$

7: $i_{R1} = u_{R1} / R_1$

8: $v_2 = u_C$

9: $i_C = i_{R1} - i_{R2}$

10: $du_C/dt = i_C / C_1$

11: $u_{R2} = u_C$

12: $i_{R2} = u_{R2} / R_2$

$\Rightarrow$

$$S = \begin{array}{c} \\ 01 \\ 02 \\ 03 \\ 04 \\ 05 \\ 06 \\ 07 \\ 08 \\ 09 \\ 10 \\ 11 \\ 12 \end{array} \begin{array}{cccccccccccc} U_0 & i_0 & u_L & \frac{di_L}{dt} & v_1 & u_{R1} & i_{R1} & v_2 & i_C & \frac{du_C}{dt} & u_{R2} & i_{R2} \end{array}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

The Structure Digraph

- The ***structure digraph*** contains the same information as the structure incidence matrix. The information is only represented differently.
- The structure digraph enumerates the equations to the left and the unknowns to the right. A connecting line between an equation and an unknown indicates that the unknown appears in the equation.

The Structure Digraph: An Example

1: $U_0 = f(t)$

2: $i_0 = i_L + i_{R1}$

3: $u_L = U_0$

4: $di_L/dt = u_L / L_1$

5: $v_1 = U_0$

6: $u_{R1} = v_1 - v_2$

7: $i_{R1} = u_{R1} / R_1$

8: $v_2 = u_C$

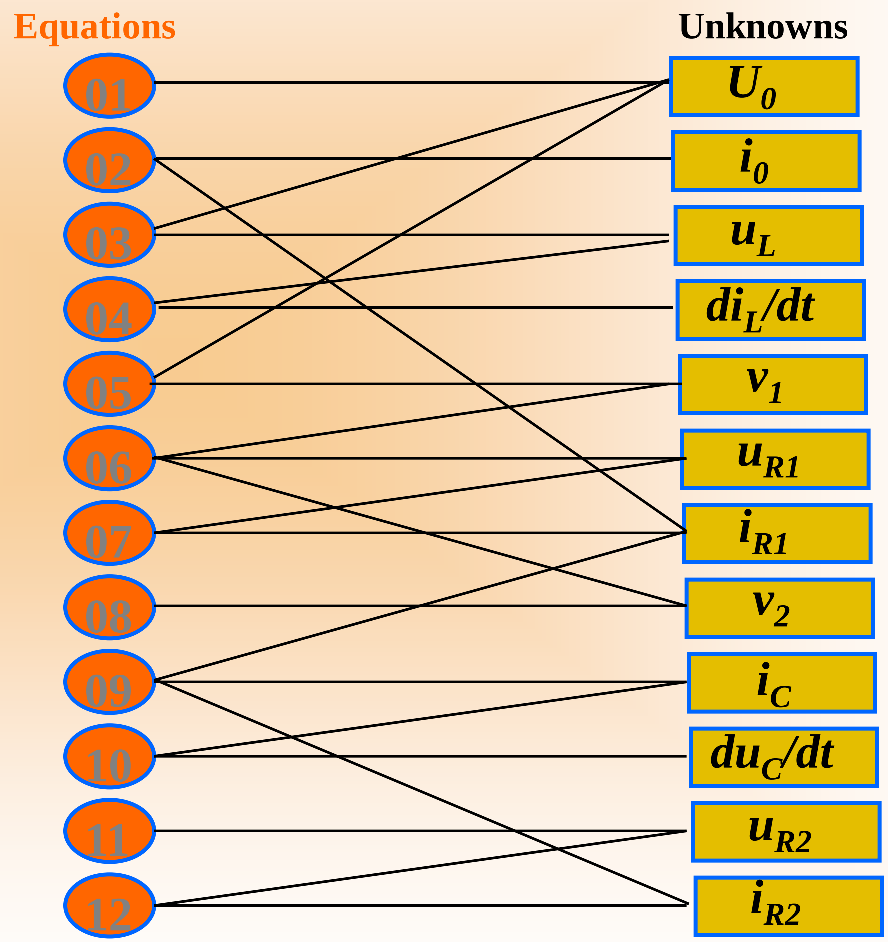
9: $i_C = i_{R1} - i_{R2}$

10: $du_C/dt = i_C / C_1$

11: $u_{R2} = u_C$

12: $i_{R2} = u_{R2} / R_2$

$\Rightarrow$



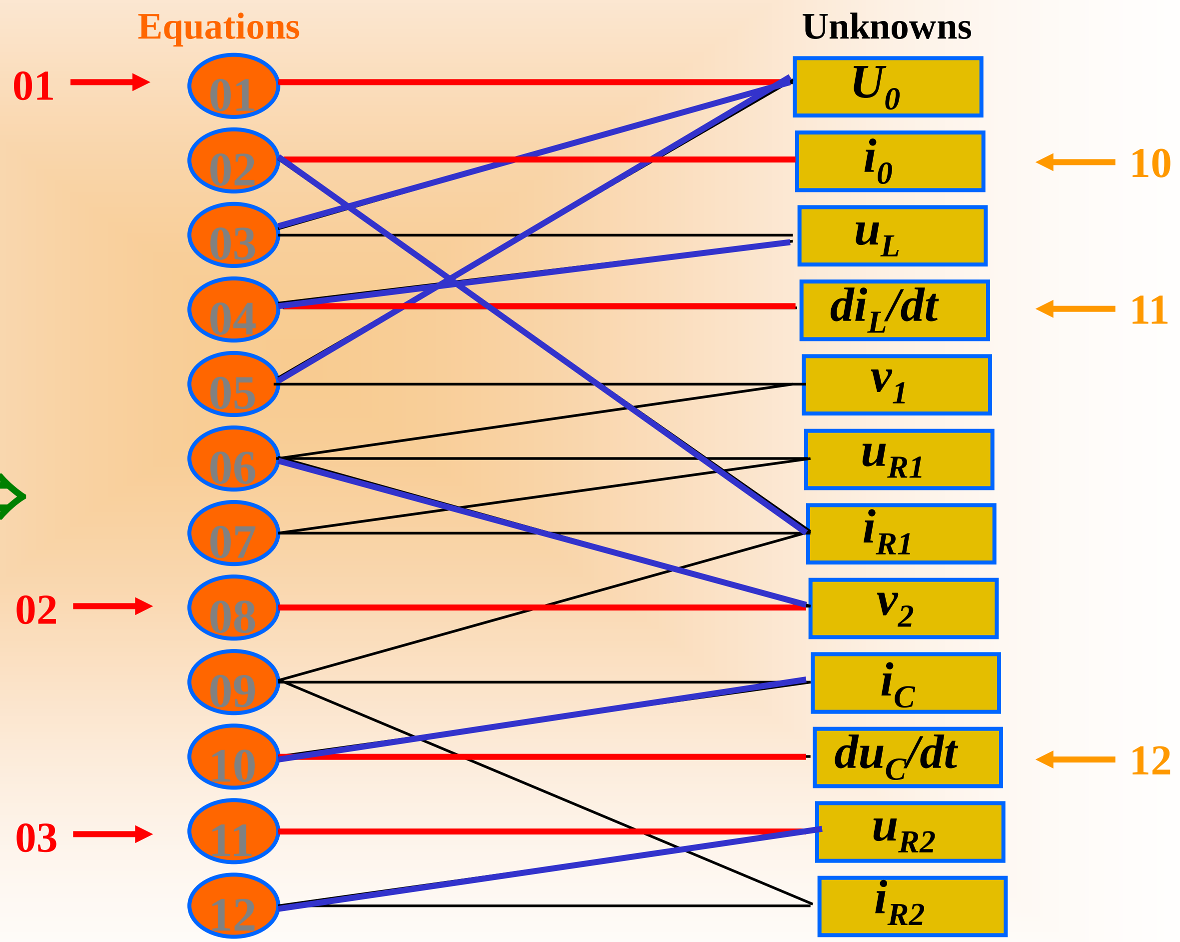
The Algorithm by Tarjan

- The ***algorithm by Tarjan*** is based on the structure digraph.
- It consists in a procedure by which the digraph is being colored.
  - ∀ equations with only one black line attached to them, that line is colored in red, and all black lines emanating from the indicated variable are colored in blue. Equations associated with lines that are freshly colored in red are renumbered in increasing order starting with ***1***.
  - ∀ unknowns with only one black line attached to them, that line is colored in red, and all black lines emanating from the indicated equation are colored in blue. Equations associated with lines that are freshly colored in red are renumbered in decreasing order starting with ***n***, the number of equations.

The Tarjan Algorithm: An Example I

1: $U_0 = f(t)$
2: $i_0 = i_L + i_{R1}$
3: $u_L = U_0$
4: $di_L/dt = u_L / L_1$
5: $v_1 = U_0$
6: $u_{R1} = v_1 - v_2$
7: $i_{R1} = u_{R1} / R_1$
8: $v_2 = u_C$
9: $i_C = i_{R1} - i_{R2}$
10: $du_C/dt = i_C / C_1$
11: $u_{R2} = u_C$
12: $i_{R2} = u_{R2} / R_2$

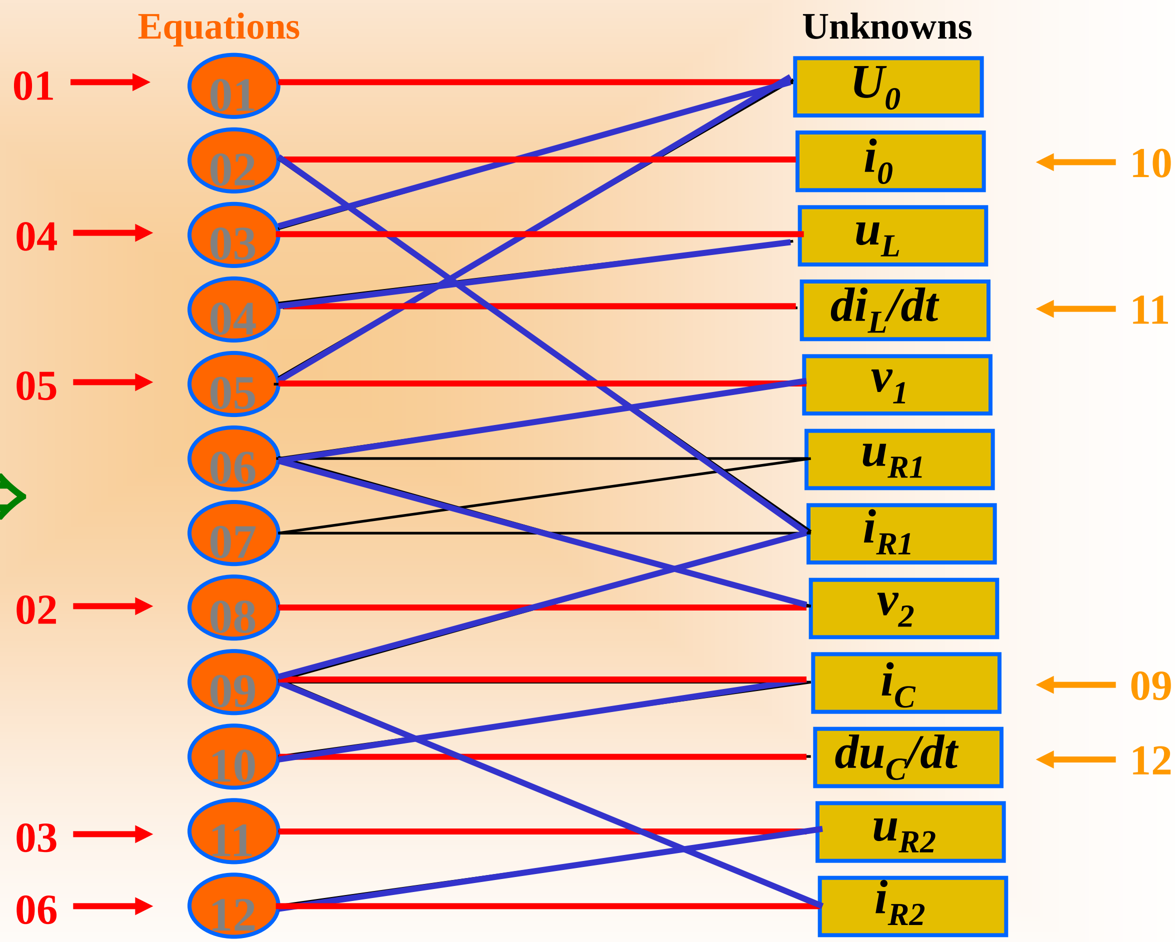
$\Rightarrow$



The Tarjan Algorithm: An Example II

1: $U_0 = f(t)$
2: $i_0 = i_L + i_{R1}$
3: $u_L = U_0$
4: $di_L/dt = u_L / L_1$
5: $v_1 = U_0$
6: $u_{R1} = v_1 - v_2$
7: $i_{R1} = u_{R1} / R_1$
8: $v_2 = u_C$
9: $i_C = i_{R1} - i_{R2}$
10: $du_C/dt = i_C / C_1$
11: $u_{R2} = u_C$
12: $i_{R2} = u_{R2} / R_2$

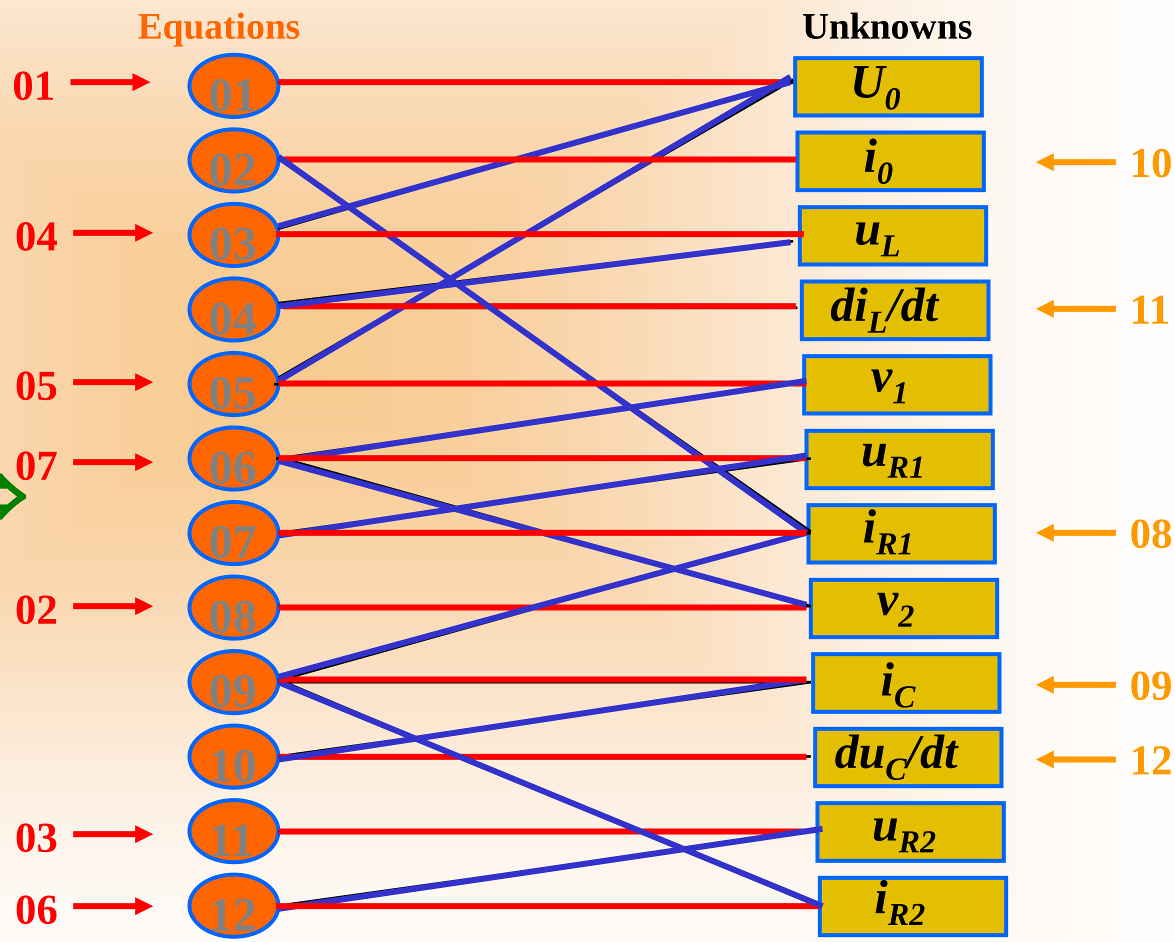
$\Rightarrow$



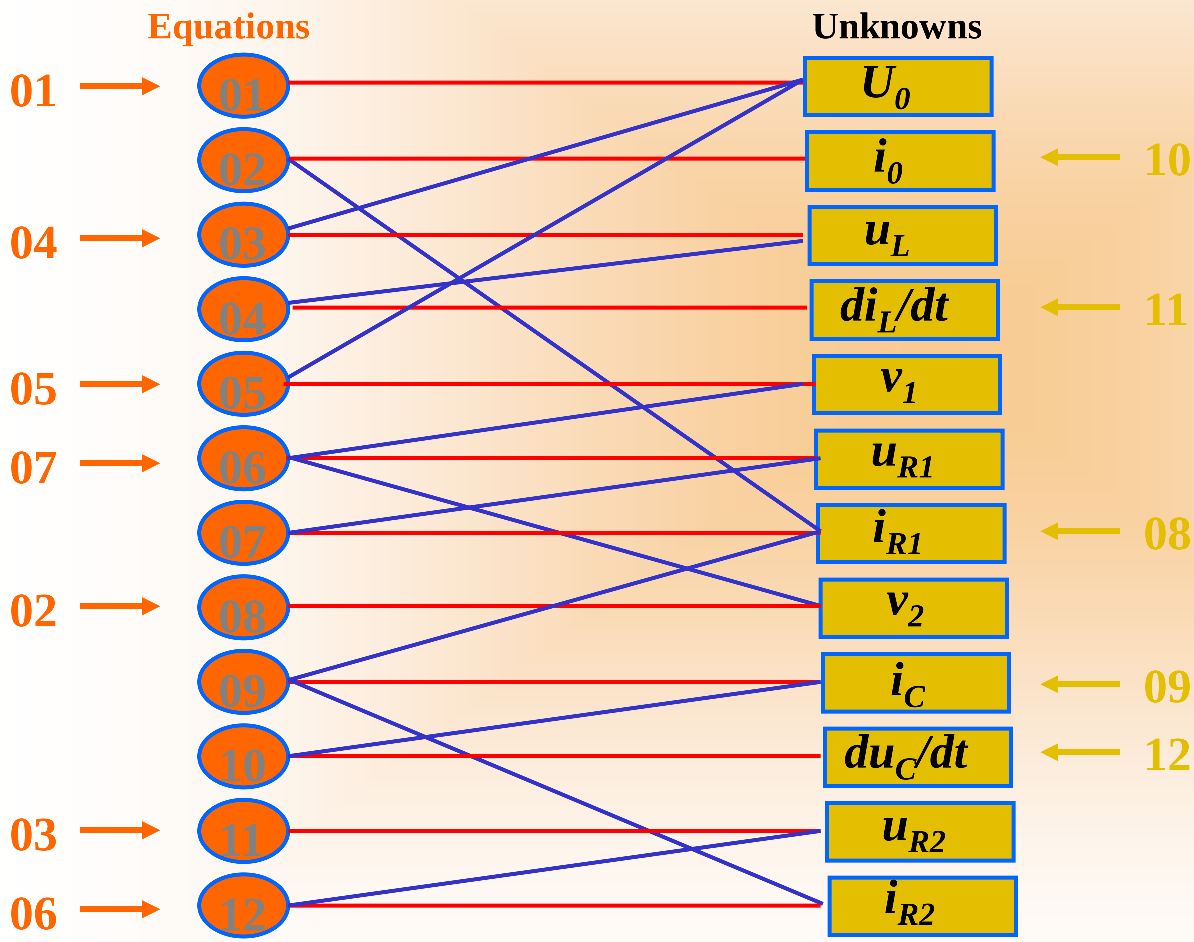
The Tarjan Algorithm: An Example III

1: $U_0 = f(t)$
2: $i_0 = i_L + i_{R1}$
3: $u_L = U_0$
4: $di_L/dt = u_L / L_1$
5: $v_1 = U_0$
6: $u_{R1} = v_1 - v_2$
7: $i_{R1} = u_{R1} / R_1$
8: $v_2 = u_C$
9: $i_C = i_{R1} - i_{R2}$
10: $du_C/dt = i_C / C_1$
11: $u_{R2} = u_C$
12: $i_{R2} = u_{R2} / R_2$

$\Rightarrow$



The Tarjan Algorithm: An Example IV



⇒

1: $U_0 = f(t)$
2: $v_2 = u_C$
3: $u_{R2} = u_C$
4: $u_L = U_0$
5: $v_1 = U_0$
6: $i_{R2} = u_{R2} / R_2$
7: $u_{R1} = v_1 - v_2$
8: $i_{R1} = u_{R1} / R_1$
9: $i_C = i_{R1} - i_{R2}$
10: $i_0 = i_L + i_{R1}$
11: $di_L/dt = u_L / L_1$
12: $du_C/dt = i_C / C_1$

The Structure Incidence Matrix II

1: $U_0 = f(t)$
2: $v_2 = u_C$
3: $u_{R2} = u_C$
4: $u_L = U_0$
5: $v_1 = U_0$
6: $i_{R2} = u_{R2} / R_2$
7: $u_{R1} = v_1 - v_2$
8: $i_{R1} = u_{R1} / R_1$
9: $i_C = i_{R1} - i_{R2}$
10: $i_0 = i_L + i_{R1}$
11: $di_L/dt = u_L / L_1$
12: $du_C/dt = i_C / C_1$

⇒

$S =$

	U_0	v_2	u_{R2}	u_L	v_1	i_{R2}	u_{R1}	i_{R1}	i_C	i_0	$\frac{di_L}{dt}$	$\frac{du_C}{dt}$
01	1	0	0	0	0	0	0	0	0	0	0	0
02	0	1	0	0	0	0	0	0	0	0	0	0
03	0	0	1	0	0	0	0	0	0	0	0	0
04	1	0	0	1	0	0	0	0	0	0	0	0
05	1	0	0	0	1	0	0	0	0	0	0	0
06	0	0	1	0	0	1	0	0	0	0	0	0
07	0	1	0	0	1	0	1	0	0	0	0	0
08	0	0	0	0	0	0	1	1	0	0	0	0
09	0	0	0	0	0	1	0	1	1	0	0	0
10	0	0	0	0	0	0	0	1	0	1	0	0
11	0	0	0	1	0	0	0	0	0	0	1	0
12	0	0	0	0	0	0	0	0	1	0	0	1

⇒ *The structure incidence matrix of the completely sorted equation system is a matrix in lower triangular (LT) form.*

Algebraic Loops: An Example I

1: $U_0 = f(t)$

2: $u_1 = R_1 \cdot i_1$

3: $u_2 = R_2 \cdot i_2$

4: $u_3 = R_3 \cdot i_3$

5: $u_L = L \cdot di_L/dt$

6: $i_0 = i_1 + i_L$

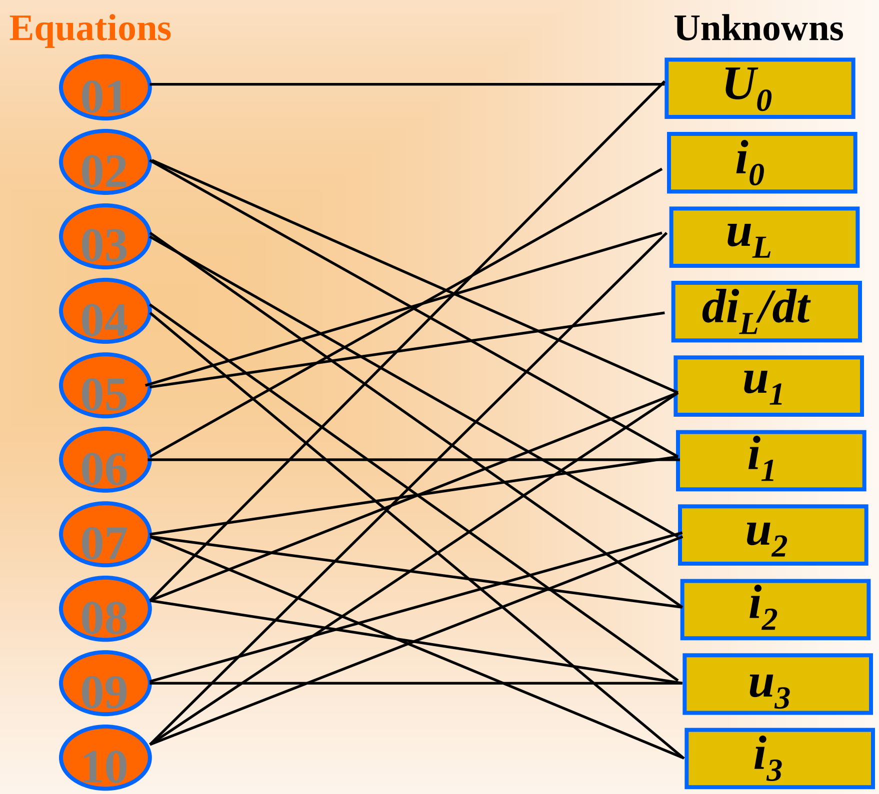
7: $i_1 = i_2 + i_3$

8: $U_0 = u_1 + u_3$

9: $u_3 = u_2$

10: $u_L = u_1 + u_2$

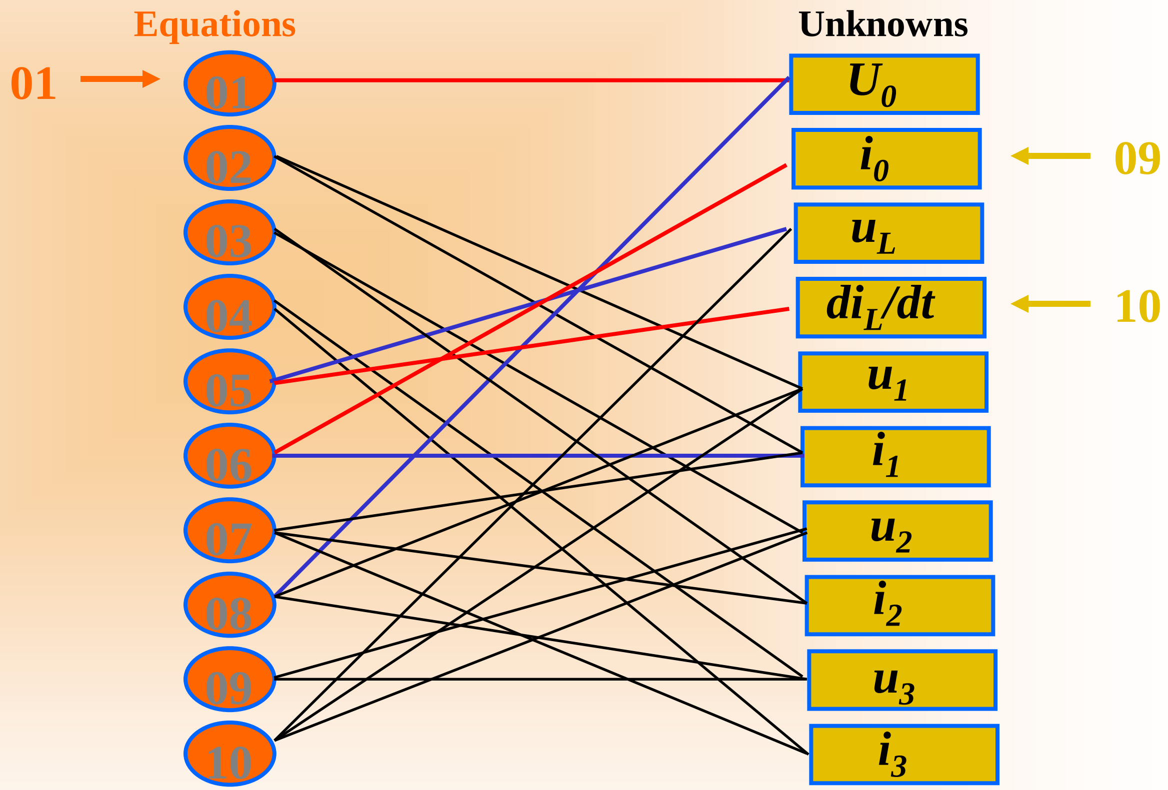
$\Rightarrow$



Algebraic Loops: An Example II

1: $U_0 = f(t)$
2: $u_1 = R_1 \cdot i_1$
3: $u_2 = R_2 \cdot i_2$
4: $u_3 = R_3 \cdot i_3$
5: $u_L = L \cdot di_L/dt$
6: $i_0 = i_1 + i_L$
7: $i_1 = i_2 + i_3$
8: $U_0 = u_1 + u_3$
9: $u_3 = u_2$
10: $u_L = u_1 + u_2$

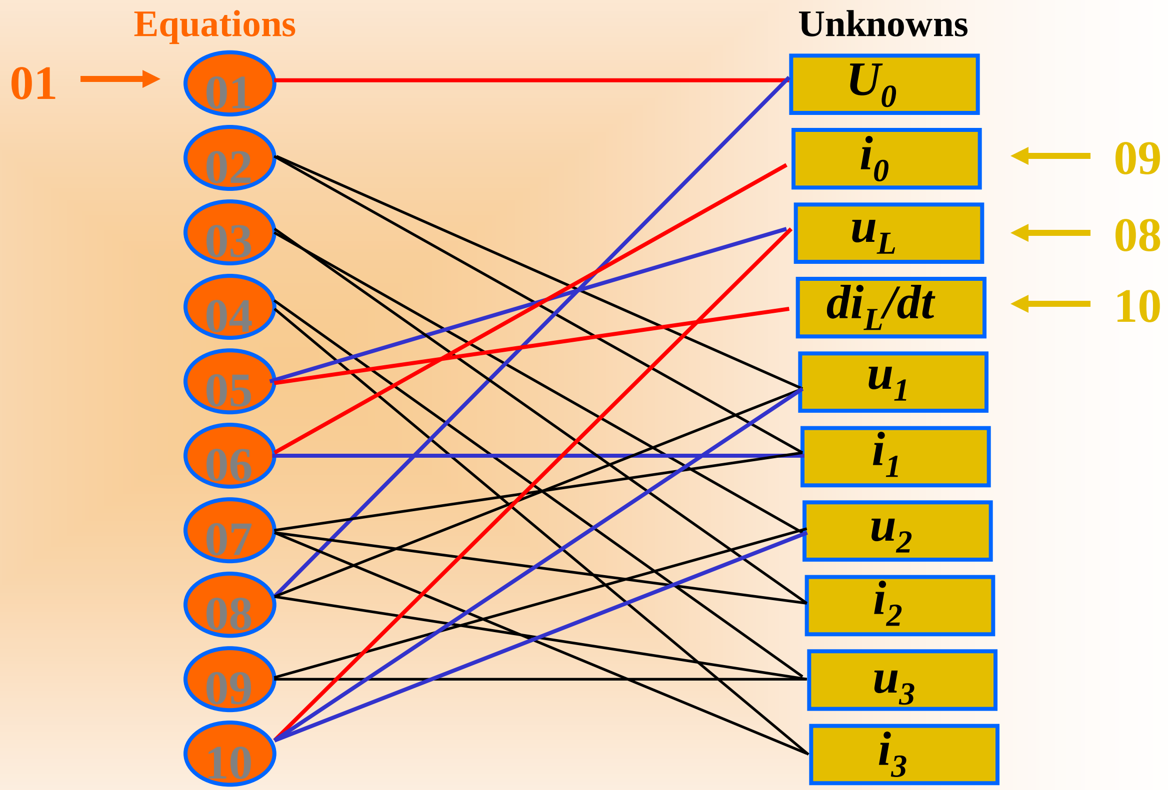
⇒



Algebraic Loops: An Example III

1: $U_0 = f(t)$
2: $u_1 = R_1 \cdot i_1$
3: $u_2 = R_2 \cdot i_2$
4: $u_3 = R_3 \cdot i_3$
5: $u_L = L \cdot di_L/dt$
6: $i_0 = i_1 + i_L$
7: $i_1 = i_2 + i_3$
8: $U_0 = u_1 + u_3$
9: $u_3 = u_2$
10: $u_L = u_1 + u_2$

⇒



⇒ *The algorithm stalls, because there are no more single black lines attached to either equations or variables.*

The Tearing of Algebraic Loops I

- The following heuristics may be used to determine suitable ***tearing variables***:
  - In the digraph, determine the equations with the largest number of black lines attached to them.
  - For every one of these equations, follow its black lines and determine those variables with the largest number of black lines attached to them.
  - For every one of these variables, determine how many additional equations can be made causal if that variable is assumed to be known.
  - Choose one of those variables as the next tearing variable that allows the largest number of additional equations to be made causal.

The Tearing of Algebraic Loops II

- In the example at hand, ***equation #7*** has 3 black lines attached. All other not yet renumbered equations only have two black lines attached.
- Equation #7 points at variables i_1, i_2, and i_3.
- Each of these variables has one additional black line attached to it.
- Variable i_1 permits to make causal all additional equations.
- Consequently, i_1 shall be used as ***tearing variable***.

Algebraic Loops: An Example IV

1: $U_0 = f(t)$

2: $u_1 = R_1 \cdot i_1$

3: $u_2 = R_2 \cdot i_2$

4: $u_3 = R_3 \cdot i_3$

5: $u_L = L \cdot di_L/dt$

6: $i_0 = i_1 + i_L$

7: $i_1 = i_2 + i_3$

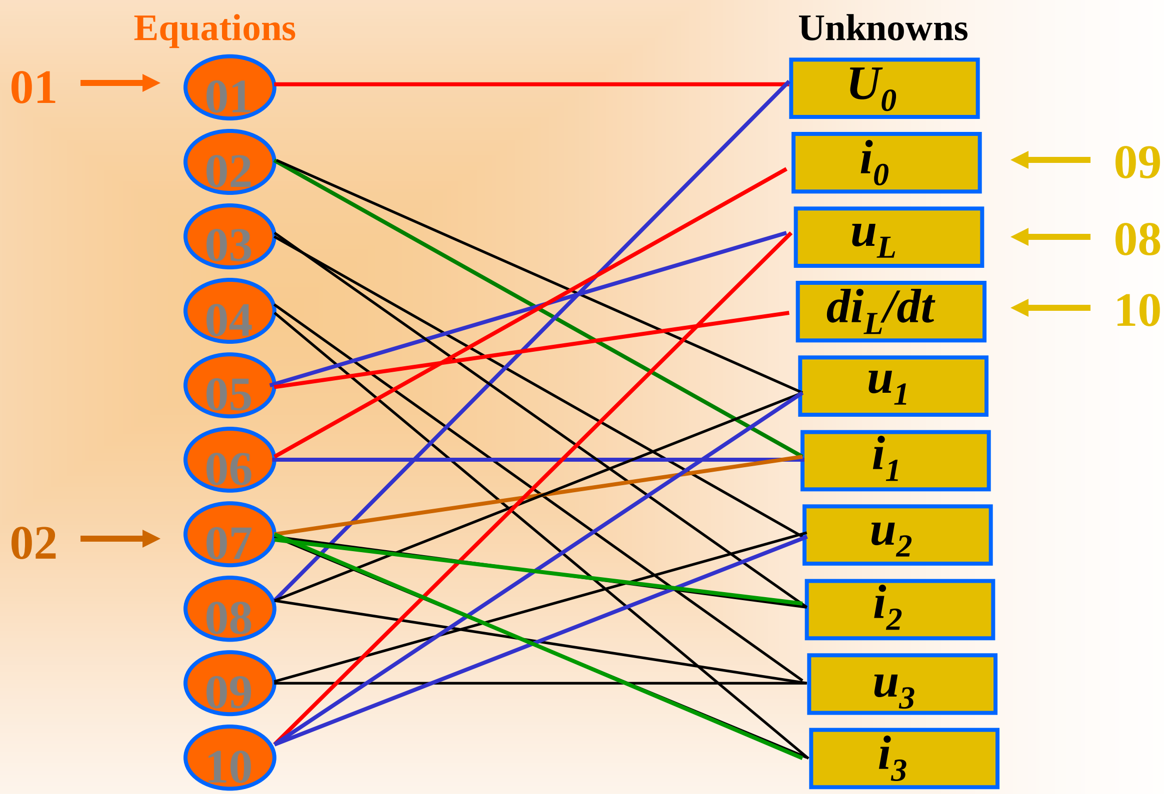
8: $U_0 = u_1 + u_3$

9: $u_3 = u_2$

10: $u_L = u_1 + u_2$

Choice

$\Rightarrow$



Algebraic Loops: An Example V

1: $U_0 = f(t)$

2: $u_1 = R_1 \cdot i_1$

3: $u_2 = R_2 \cdot i_2$

4: $u_3 = R_3 \cdot i_3$

5: $u_L = L \cdot di_L/dt$

6: $i_0 = i_1 + i_L$

7: $i_1 = i_2 + i_3$

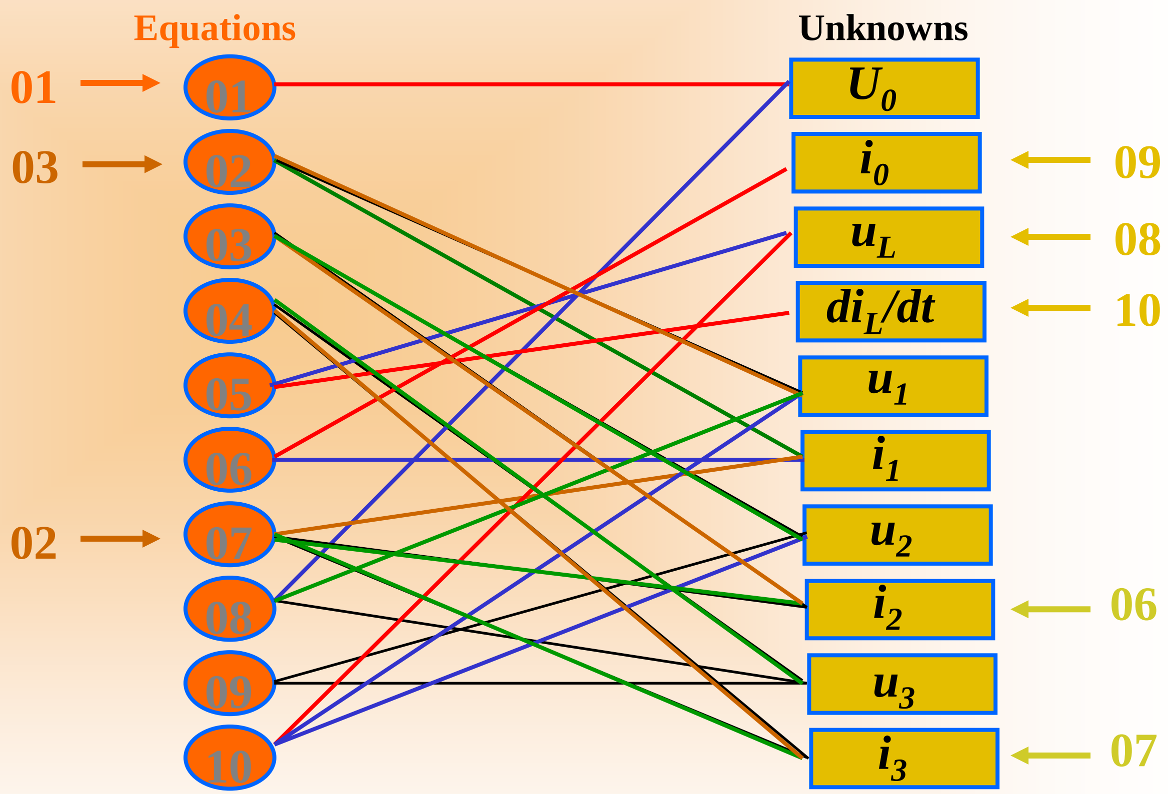
8: $U_0 = u_1 + u_3$

9: $u_3 = u_2$

10: $u_L = u_1 + u_2$

Choice

⇒



Algebraic Loops: An Example VI

1: $U_0 = f(t)$

2: $u_1 = R_1 \cdot i_1$

3: $u_2 = R_2 \cdot i_2$

4: $u_3 = R_3 \cdot i_3$

5: $u_L = L \cdot di_L/dt$

6: $i_0 = i_1 + i_L$

7: $i_1 = i_2 + i_3$

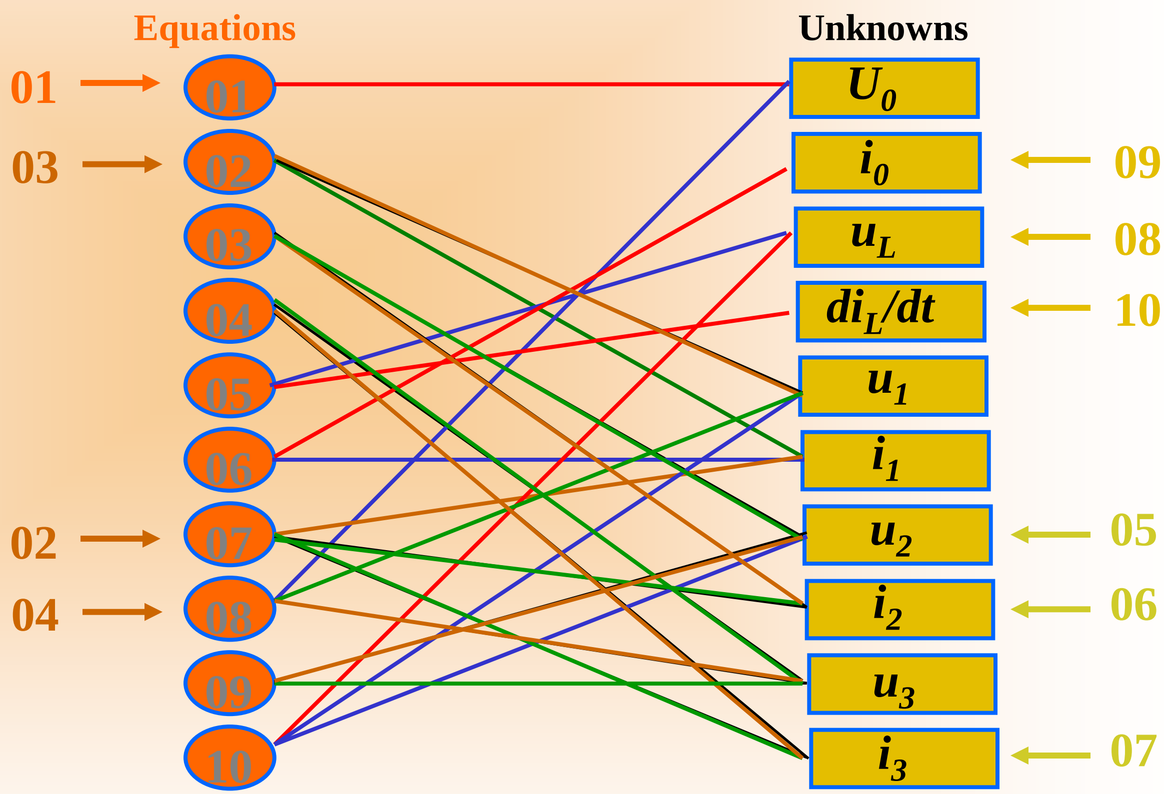
8: $U_0 = u_1 + u_3$

9: $u_3 = u_2$

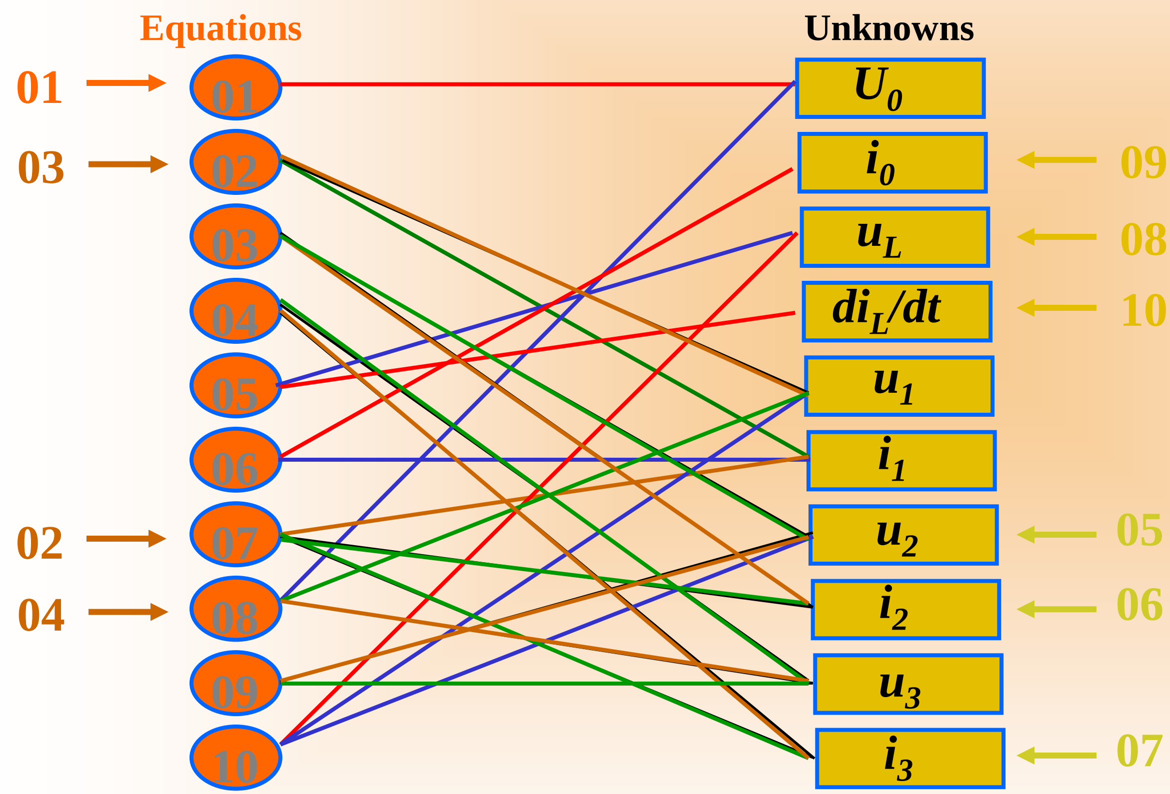
10: $u_L = u_1 + u_2$

Choice

$\Rightarrow$



Algebraic Loops: An Example VII



⇒

$$1: \quad U_0 = f(t)$$
$$2: \quad i_1 = i_2 + i_3$$
$$3: \quad u_1 = R_1 \cdot i_1$$
$$4: \quad u_3 = U_0 - u_1$$
$$5: \quad u_2 = u_3$$
$$6: \quad i_2 = u_2 / R_2$$
$$7: \quad i_3 = u_3 / R_3$$
$$8: \quad u_L = u_1 + u_2$$
$$9: \quad i_0 = i_1 + i_L$$
$$10: \quad di_L/dt = u_L / L$$

Choice

The Structure Incidence Matrix III

1: $U_0 = f(t)$

2: $i_1 = i_2 + i_3$

3: $u_1 = R_1 \cdot i_1$

4: $u_3 = U_0 - u_1$

5: $u_2 = u_3$

6: $i_2 = u_2 / R_2$

7: $i_3 = u_3 / R_3$

8: $u_L = u_1 + u_2$

9: $i_0 = i_1 + i_L$

10: $di_L/dt = u_L / L$

Choice

$\Rightarrow$

$S =$

	U_0	i_1	u_1	u_3	u_2	i_2	i_3	u_L	i_0	$\frac{di_L}{dt}$
01	1	0	0	0	0	0	0	0	0	0
02	0	1	0	0	0	1	1	0	0	0
03	0	1	1	0	0	0	0	0	0	0
04	1	0	1	1	0	0	0	0	0	0
05	0	0	0	1	1	0	0	0	0	0
06	0	0	0	0	1	1	0	0	0	0
07	0	0	0	1	0	0	1	0	0	0
08	0	0	1	0	1	0	0	1	0	0
09	0	1	0	0	0	0	0	0	1	0
10	0	0	0	0	0	0	0	1	0	1

$\Rightarrow$ *The structure incidence matrix assumes the form of a block lower triangular (BLT) matrix.*

The Solving of Algebraic Loops I

- The ***Tarjan algorithm*** thus identifies and isolates algebraic loops.
- It transforms the ***structure incidence matrix*** to ***BLT form***, whereby the diagonal blocks are made as small as possible.
- The selection of the ***tearing variables*** is not done in a truly optimal fashion. This is not meaningful, because the optimal selection of tearing variables has been shown to be an ***np-complete problem***. Instead, a set of heuristics is being used, which usually comes up with a small number of tearing variables, although the number may not be truly minimal.
- The ***Tarjan algorithm*** does not concern itself with how the resulting ***algebraic loops*** are being solved.

The Solving of Algebraic Loops II

- The ***algebraic loops*** can be solved either ***analytically*** or ***numerically***.
- If the loop equations are ***non-linear***, a ***Newton iteration on the tearing variables*** may be optimal.
- If the loop equations are ***linear*** and if the set is fairly large, ***Newton iteration*** may still be the method of choice.
- If the loop equations are linear and if the set is of modest size, the equations can either be solved by ***matrix techniques*** or by means of explicit ***formulae manipulation***.
- The ***Modelica*** modeling environment uses a set of appropriate heuristics to select the best technique automatically in each case.

References

- Elmqvist H. and M. Otter (1994), "Methods for tearing systems of equations in object-oriented modeling," *Proc. European Simulation Multiconference*, Barcelona, Spain, pp. 326-332.
- Tarjan R.E. (1972), "Depth first search and linear graph algorithms," *SIAM J. Comp.*, **1**, pp. 146-160.