Записки Ленинградского ордена Ленина, ордена Октябрьской Революции
и ордена Трудового Красного Знамени горного института им. Г.В.Плеханова,
т. XCIV, 1982 г.

УДК 621.314.58

АНАЛИТИЧЕСКОЕ ИССЛЕДОВАНИЕ СИСТЕМ ТИРИСТОРНОГО ЭЛЕКТРОПРИВОДА ПРИ ПОМОЩИ ЭВМ

В.А.ДАРТАУ

Подготовка системы дифференциальных уравнений электромеханических систем, а тем более процесс преобразований к удобному для решения на ЭВМ виду, обычно занимает гораздо больше времени, чем программирование и решение (вычисление) ЭВМ. Однако этот вопрос получил в настоящее время очень слабое развитие.

Процесс подготовки систем уравнений можно разделить на этапы, где требуется высокая квалификация исследователя, и этапы рутинных аналитических выкладок и преобразований.

К первым относятся: 1) проектирование схемы исследуемого устройства (привода); 2) выбор фундаментальной системы переменных (для электромеханических систем – система фундаментальных контуров); 3) выбор и обоснование преобразований системы к удобному для численного решения виду.

Ко вторым относятся: 1) формальное составление системы дифференциальных уравнений; 2) составление матриц и функций преобразования; 3) проведение преобразований (в простейшем случае – умножение матриц); 4) преобразования, упрощение формул [2].

Все эти этапы для достаточно больших систем являются (даже по числу элементов) очень трудоемкими и, что важно, трудно контролируемыми, но могут быть выполнены при помощи вычислительной машины, что чрезвычайно повысит производительность исследований.

Машинная реализация этих этапов аналитических манипуляций может быть основана либо на теории графов, либо на тензорном анализе сетей. Мы будем придерживаться последнего, как более физичного [3].

Набор многополюсников, входящих в систему, с явно заданными функциями связи между переменными состояния и входными переменными будем называть примитивной сетью. В случае чисто электромагнитной системы функциями связи являются зависимости между токами и разностями потенциалов многополюсника.

В качестве **представления примитивной сети** возьмем матрицу, элементами которой являются операторы, **связывающие** i-ю переменную состояния с j-й переменной состояния (по терминологии Крона – матрица импеданса контурной сети). **Тогда** примитивная контурная сеть будет описываться векторной системой операторных уравнений

$$Z(I) = E,$$

где Z – матричная функция импедансов, элементами которой являются операторы (например, $Z_{ij} = (\cdot)^2$, иначе влияние j-й переменной на i-ю определено как $(J_j)^2$). Следует отметить, что переменные могут быть любой физической природы, название импеданс чисто условное.

Элементы примитивной сети в конкретную схему объединяются при помощи матрицы соединений, которая строится по уравнениям непрерывности в узлах. Собранная сеть характеризуется новой системой контуров, причем число и конкретное перечисление фундаментальных контуров неизвестно. Выбор конкретного набора фундаментальных контуров имеет первостепенное значение для успешного численного расчета, так как при этом определяются независимые переменные сети и структура системы дифференциальных уравнений. Перечисление всех систем фундаментальных контуров ручным способом для достаточно больших сетей представляется практически невозможным.

Однако эта задача легко решается при помощи следующего алгоритма:

1) все ветви помещаются в хранилище *SETV* и упорядочиваются согласно переменным весам, заданным разработчиком;

2) все узлы помещаются в хранилище *SETY* и нумеруются в произвольном порядке;

3) вводится множество корней деревьев *LES*;

4) из хранилища *SETV* берется очередная ветвь;

5) если оба узла этой ветви находятся в хранилище *SETY*, то берется первое свободное место в множестве *LES* и присоединяется узел, к этому узлу присоединяется ветвь и к ней – второй узел, передача на № 8;

6) если один узел в хранилище *SETY*, а второй отсутствует (он уже в лесу), то ищется корень дерева в множестве *LES* и, опускаясь по ветвям, ищется отсутствующий узел, к нему присоединяется ветвь и к ней узел из *SETY*, передача на № 8;

7) если оба узла в хранилище *SETY* отсутствуют, то, если узлы находятся в разных деревьях, одно из деревьев отсоединяется от множества и при помощи ветви присоединяется к другому; в противном случае ветвь остается в хранилище *SETV*;

8) если в хранилище *SETV* еще остаются не пройденные ветки, то переход на № 4;

9) КОНЕЦ.

Этот алгоритм строит **фундаментальное** дерево, причем в зависимости от упорядоченности ветвей получаются разные деревья, а следовательно, и контуры [1]. Замыкающие ветви сохраняются в хранилище *SETV*, переменные этих ветвей являются переменными состояния.

Следующей задачей является построение матрицы преобразования примитивной контурной сети, фундаментальное дерево которой построено.

Матрица преобразования строится по следующему алгоритму:

1) проходя по дереву от корня вниз, находим свободный узел (лист дерева); этот узел характеризуется тем, что к нему присоединена только одна ветвь, все остальные ветви, которые

присоединены к сети, отсутствуют и, следовательно, являются замыкающими; по алгебраическая сумма всех переменных узла равна нулю – следовательно, из этого уравнения получаем выражение для переменной присоединенной ветви;

2) узел и ветвь удаляются в хранилище *SETV* и *SETV*;

3) если остался один узел, то переход на КОНЕЦ, иначе переход на № 1.

Алгоритм строит матрицу с целочисленными алементами размером $M \times N$ (M – число ветвей или контуров в примитивной сети; N – число фундаментальных контуров, независимых переменных).

Операторная матрица сети имеет вид

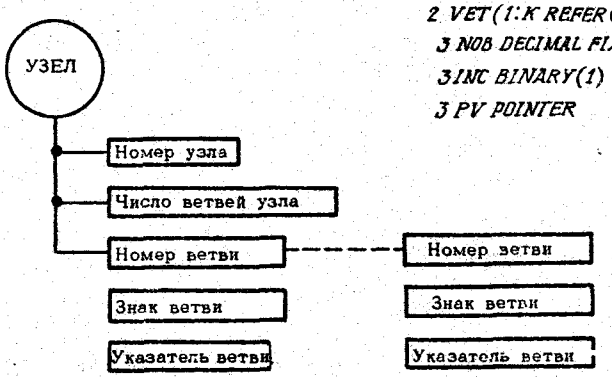
$$F(I_\phi) = MNC^T Z(MNC \times I_\phi),$$

где MNC – матрица преобразования; I_ϕ – вектор фундаментальных переменных.

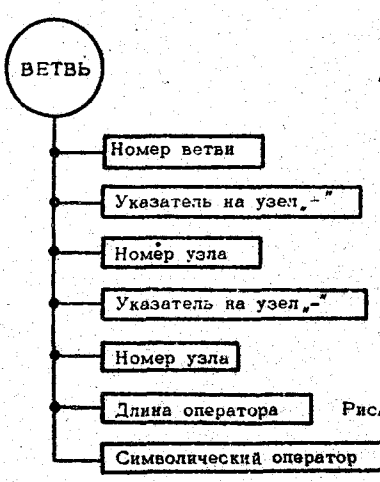
Программирование проводилось на языке ПЛ/I, узлы и ветви определялись как базированные структуры, что позволило применить технику управления

```
DCL 1 YZEL BAZED(PY),
      2 NO DECIMAL FIXED,
      2 L BINARY FIXED,
      2 VET(1:K REFER(YZEL.L)),
       3 NOB DECIMAL FIXED,
       3 INC BINARY(1)
       3 PV POINTER
```

а)



б)




```
DCL 1 REBR BASED (PV),
      2 NR DECIMAL FIXED,
      2 PU POINTER,
      2 NU DECIMAL FIXED,
      2 PD POINTER,
      2 ND DECIMAL FIXED,
      2 LR BINARY FIXED,
      2 INF CHAR (M REFER(REBR.LR))
```

Рис.1. Описание узлов (а) и ветвей (б) сети

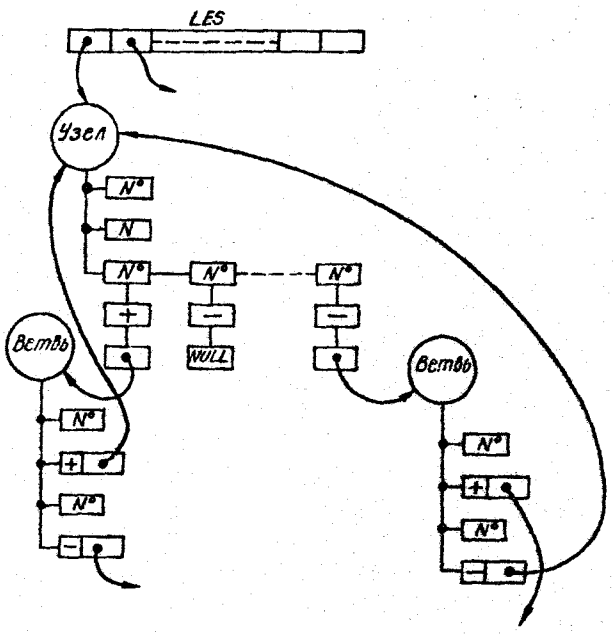
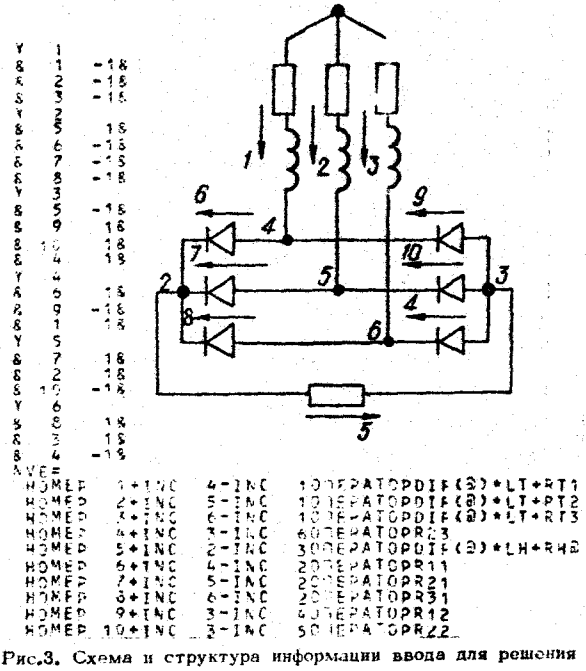


Рис.2. Структура дерева сети




```
Y   1
&   1   -18
&   2   -15
&   3   -15
Y   2
&   5    18
&   6   -15
&   7   -15
&   8   -18
Y   3
&   5   -15
&   9    18
&  10    18
&   4    18
Y   4
&   6    18
&   9   -18
&   1   -18
Y   5
&   7    18
&   2    18
&  10   -18
Y   6
&   8    18
&   3    18
&   4   -18
AVE=
  HOMER   1+INC   4-INC   1OPERATOPDIF(3)*LT+RT1
  HOMER   2+INC   5-INC   1OPERATOPDIF(@)*LT+PT2
  HOMER   3+INC   6-INC   1OPERATOPDIF(@)*LT+RT3
  HOMER   4+INC   1-INC   6OPERATOPR23
  HOMER   5+INC   2-INC   3OPERATOPDIF(3)*LH+RH3
  HOMER   6+INC   4-INC   2OPERATOPR11
  HOMER   7+INC   5-INC   2OPERATOPR21
  HOMER   8+INC   6-INC   2OPERATOPR31
  HOMER   9+INC   3-INC   4OPERATOPR12
  HOMER  10+INC   3-INC   5OPERATOPR22
```

Рис.3. Схема и структура информации ввода для решения задачи

списками. Определение структур приведено на рис.I.

В структурах фигурируют знаки ветвей и узлов, которые определяют знак вхождения переменной ветви в присоединенный узел. Связи узлов и ветвей в дереве показаны на рис.2. На рис.2 показан корневой узел, связанный с несколькими ветвями. Первая из ветвей присоединена знаком „+", вторая в дерево не вошла и третья присоединена знаком „−".

На рис.3 приведена схема тиристорного выпрямителя, показана нумерация узлов и ветвей, стрелками показаны положительные вхождения токов в узлы (стрелка показывает на узел, имеющий знак „+").

Ниже приведены результаты решения - построено три варианта деревьев и матриц преобразования.

Вариант I

Весовые коэффициенты ветвей

```
      N=  1 => 3.000E+01 N=  2 => 3.000E+01 N=  3 => 3.000E+01 N=  4 => 4.000E+01
N=  4 => 2.000E+01 N=  9 => 2.000E+01 N= 10 => 2.000E+01 N=  7 => 4.000E+01
N=  6 => 1.000E+01 N=  8 => 1.000E+01
```

Перечисление дерева

Дерево

```
     2     4     1     6     3   MAX=     4
                 !->   6!->   3
 !->   5!->   2!->   4!->   1
ЗАМЫКАЮЩИЕ ВЕТВИ
     2     5     1  DIF(W)*LT+R12
     3     6     1  DIF(W)*LT+R13
     5     2     3  DIF(W)*LM+RHW
     9     5     4  R12
    10     3     5  R22
 TIL=
        2           3           5           9          10
```

Топологическая матрица

```
 -1   -1    0    0    0
  1    0    0    0    0
  0    1    0    0    0
  0    0    1   -1   -1
  0    0    1    0    0
  1    1    0    1    0
 -1    0    0    0    1
  0   -1    1   -1   -1
  0    0    0    1    0
  0    0    0    0    1
```

Матрица системы для фундаментальных переменных

```
1      1  -(-(R21))+((R11))+((DIF(W)*LI+R12))-(-(DIF(W)*LI+RT))
1      2  ((R11))-(-(DIF(W)*LI+RT))
1      4  ((R11))
1      5  -((R21))
2      1  ((R11))-(-(DIF(W)*LI+R11))
2      2  -(-(R31))+((R11))+((DIF(W)*LI+R13))-(-(DIF(W)*LI+RT))
2      3  -((R31))
2      4  -(-(R31))+((R11))
2      5  -(-(R31))
3      2  (-(R31))
3      3  ((R31))+((DIF(W)*LM+RHW))+((R23))
3      4  (-(R31))+(-(R23))
3      5  (-(R31))+(-(R23))
4      1  ((R11))
4      2  -(-(R31))+((R11))
4      3  -((R31))-((R23))
4      4  ((R12))-(-(R31))+((R11))-(-(R23))
4      5  -(-(R31))-(-(R23))
5      1  (-(R21))
5      2  -(-(R31))
5      3  -((R31))-((R23))
5      4  -(-(R31))-(-(R23))
5      5  ((R22))-(-(R31))+((R21))-(-(R23))
```

Вариант 2

Весовые коэффициенты ветвей

```
      LOSI
  N=  1 => 5.000E+01 N=  4 => 4.000E+01 N=  9 => 4.000E+01 N= 10 => 4.000E+01 N=  5 => 4.000E+01
N=  7 => 2.000E+01 N=  8 => 2.000E+01 N=  2 => 6.000E+01 N=  3 => 5.000E+01
N=  6 => 2.000E+01
```

Перечисление дерева

Дерево

```
     2     3     5     6     1   MAX=     3
        !=> 1
              !=>  6
              !=>  5
  !=>    4!=>    2!=>  3
 ЗАМЫКАЮЩИЕ ВЕТВИ
     2     3     1  DIF(@)*LI+R12
     3     6     1  DIF(@)*LI+R13
     4     3     6  R23
     9     3     4  R12
    10     3     5  R22
 TIL=
        2     3     4     9    10
```

Топологическая матрица

```
-1   -1    0    0    0
 1    0    0    0    0
 0    1    0    0    0
 0    0    1    0    0
 0    0    1    1    1
-1    1    0    1    0
 1    0    0    0    1
 0   -1    1    0    0
 0    0    0    1    0
 0    0    0    0    1
```

Матрица системы для фундаментальных переменных

```
1     1  -(-(R21))+((R11))+((DIF(@)*LI+R12))-(-(DIF(@)*LI+RT1))
1     2  ((R11))-(-(DIF(@)*LI+RT1))
1     4  ((R11))
1     5  -((R21))
2     1  ((R11))-(-(DIF(@)*LI+RT1))
2     2  -(-(R31))+((R11))+((DIF(@)*LI+R13))-(-(DIF(@)*LI+RT1))
2     3  -((R31))
2     4  ((R11))
2     2  (-(R31))
3     3  ((R31))+((DIF(@)*LH+RH@))+((R23))
3     4  ((DIF(@)*LH+RH@))
3     5  ((DIF(@)*LH+RH@))
4     1  ((R11))
4     2  ((R11))
4     3  ((DIF(@)*LH+RH@))
4     4  ((R12))+((R11))+((DIF(@)*LH+RH@))
4     5  ((DIF(@)*LH+RH@))
5     1  (-(R21))
5     3  ((DIF(@)*LH+RH@))
5     4  ((DIF(@)*LH+RH@))
5     5  ((R22))+((R21))+((DIF(@)*LH+RH@))
```

Вариант 3

Весовые коэффициенты ветвей

```
      COS!
        N=  1 => 1.000E+00 N=  2 => 1.000E+00 N=  9 => 2.000E+00 N= 10 => 2.000E+00 N=  4 => 2.000E+00
N=  6 => 3.000E+00 N=  8 => 3.000E+00 N=  3 => 1.000E+00 N=  7 => 3.000E+00
N=  5 => 1.000E+00
```

Перечисление дерева

Дерево

```
     3     4     1     9     6   MARK      3
                                  !=>     6
 !=>    2!=>    3!=>    4!=>    1!=>     5
ЗАМЫКАЮЩИЕ ВЕТВИ
    4     3     6  R23
    6     4     2  R11
    7     5     2  R21
    8     6     2  R31
   10     3     5  R22
 TIL=
        4        6        7        8       10
```

Топологическая матрица

```
 -1    0    1    1   -1
  0    0   -1    0    1
  1    0    0   -1    0
  1    0    0    0    0
  0    1    1    1    0
  0    1    0    0    0
  0    0    1    0    0
  0    0    0    1    0
 -1    1    1    1   -1
  0    0    0    0    1
```

Матрица системы для фундаментальных переменных

```
1      1 -(-(R12))*((R23))*((DIF(@)*LT+RT3))-(-(DIF(@)*LT+RT1))
1      2 -((R12))
1      3 -((R12))-((DIF(@)*LT+RT1))
1      4 -((R12))*(-(DIF(@)*LT+RT3))-((DIF(@)*LT+RT1))
1      5 -(-(R12))-(-(DIF(@)*LT+RT1))
2      1 (-(R12))
2      2 ((R12))+((R11))+((DIF(@)*LH+RH@))
2      3 ((R12))+((DIF(@)*LH+RH@))
2      4 ((R12))+((DIF(@)*LH+RH@))
2      5 (-(R12))
3      1 (-(R12))+(-(DIF(@)*LT+RT1))
3      2 ((R12))+((DIF(@)*LH+RH@))
3      3 ((R12))+((R21))+((DIF(@)*LH+RH@))-(-(DIF(@)*LT+RT2))+((DIF(@)*LT+RT1))
3      4 ((R12))+((DIF(@)*LH+RH@))+((DIF(@)*LT+RT1))
3      5 (-(R12))-((DIF(@)*LT+RT2))+(-(DIF(@)*LT+RT1))
4      1 (-(R12))-((DIF(@)*LT+RT3))+(-(DIF(@)*LT+RT1))
4      2 ((R12))+((DIF(@)*LH+RH@))
4      3 ((R12))+((DIF(@)*LH+RH@))+((DIF(@)*LT+RT1))
4      4 ((R12))+((R31))+((DIF(@)*LH+RH@))-(-(DIF(@)*LT+RT3))+((DIF(@)*LT+RT1))
4      5 (-(R12))+(-(DIF(@)*LT+RT1))
5      1 -(-(R12))-(-(DIF(@)*LT+RT1))
5      2 -((R12))
5      3 -((R12))+(-(DIF(@)*LT+RT2))-((DIF(@)*LT+RT1))
5      4 -((R12))-((DIF(@)*LT+RT1))
5      5 ((R22))-(-(R12))+((DIF(@)*LT+RT2))-(-(DIF(@)*LT+RT1))
```

ЛИТЕРАТУРА

1. Ахо А., Хопкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов. М., Мир, 1979.

2. Кнут Д. Искусство программирования для ЭВМ. М., Мир, 1976, т. 1.

3. Крон Г. Тензорный анализ сетей. М., Советское радио, 1979.