Лекция №7. Алгоритмы на топологических моделях.

Алгоритмы на топологических моделях. Представление графов в ЭВМ. Матрицы смежности, изоморфности, достижимости и контрдостижимости, списочные формы. Алгоритмы на графах. Алгоритмы поиска путей, выделения контуров, поиск касающихся контуров.

6.1. Задачи анализа топологии

Под топологическим анализом понимается выявление структурных свойств и особенностей модели на основании исследования моделей первого ранга неопределенности Ms(1), т.е. на основании информации о взаимосвязи переменных графа.

К основным задачам анализа топологии, относятся задачи поиска путей, выделения контуров, декомпозиции на подсистемы.

Алгоритмы для такого анализа на основании представлений моделей в форме графов или матричной форме неоднократно приводились в литературе [6, 28, 107]. Традиционные постановки касаются в основном линейных систем, составленных из однонаправленных элементов.

Алгоритмы топологического анализа имеют огромное значение для исследования СС НСУ с помощью ЭВМ, так как проблема повышения эффективности по быстродействию и точности существующих методов моделирования может быть решена за счет более полного учета топологических особенностей модели [45, 46, 124, 126].

Пример вычисление передаточных функций системы по формуле Мезона.

Расписать пример.

6.2. Представление информации о топологии моделей

Представление топологии модели возможно в списочной и матричной форме. При организации программных средств чаще используется списочная форма. При больших размерностях одноуровневых сильно разряженных моделей она имеет преимущества по требуемой памяти и скорости работы алгоритмов топологического анализа. Однако для сильно связанных систем небольшой размерности или иерархических систем эффективнее испробовать алгоритмы, основанные на матричных формах, например на матрицах смежности.

В качестве иллюстрации на рис. 1.1. приведена диаграмма графа модели странного аттрактора Лоренца [93]. Эта форма представления позволяет эффективнее решать задачи выделения путей и контуров, связности, структурной управляемости и многие другие, чем в форме НФК и отчасти СНДУ.

Модель системы представляется ориентированным графом $H=<G,H>$ с множеством переменных $X=x_1, \ldots , x_n, N$ - общее множество вершин, и множеством дуг G - упорядоченных пар номеров смежных вершин (i,j), $G=(i,j)1, \ldots (i,j)n$. Общее количество таких пар обозначено в примерах как Q.

Несмотря на всю компактность и удобство такой записи, на практике чаще используют матрицу смежности $R = r_{ij}$, показывающую наличие дуги между i-ой и j-ой вершинами.

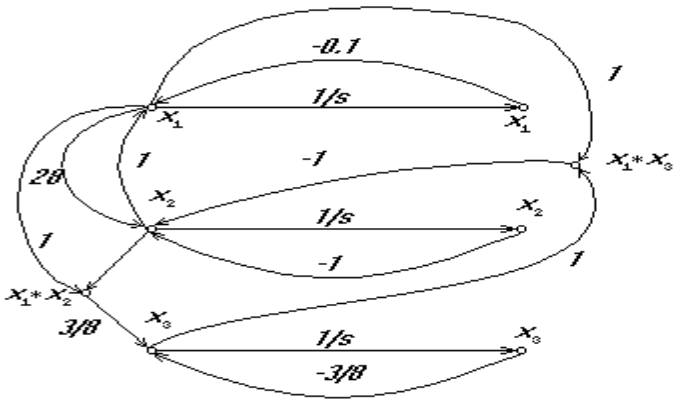


Рис. 2.1. Модель странного аттрактора в форме ориентированного графа

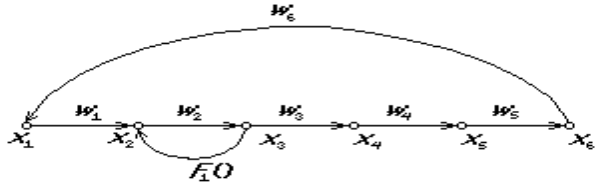


Рис. 2.2. Модель системы в форме графа

Другим способом представления топологии является матрица изоморфности **D**, в строках которой представлены номера входящих (с плюсом) и выходящих (с минусом) дуг.

Для приведенного на рис. 2.2 примера матрицы смежности и изоморфности имеют вид:

$$\mathbf{R} = \begin{vmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{vmatrix}, \quad \mathbf{D} = \begin{vmatrix} +6 & -1 \\ +1,+7 & -2 \\ +2 & -3,-7 \\ +3 & -4 \\ +4 & -5 \\ +5 & -6 \end{vmatrix}.$$

Избыточность хранимой информации в матрице смежности (нулевые значения) компенсируются простотой вычислительных алгоритмов и скоростью получения требуемой информации из матрицы. Кроме того, наличие только двух значений 0 или 1, дает возможность использовать для ее представления битовые поля, что дает значительную экономию памяти, и при размерах системы порядка 100 элементов не уступает по затратам ресурсов на хранение матрицы изоморфности, при значительно более простых алгоритмов обработки информации. Использование матриц смежности, инцидентностей, достижимостей и др. имеет большое применение для алгоритмов топологического анализа СС НСУ [107].

Ориентированные графы (структурные схемы) обычно широко используются при описании линейных систем и систем с одновходовыми нелинейностями. Однако возникают некоторые затруднения при описании нелинейных систем, где нелинейные функции могут зависеть от нескольких переменных, например при описании операций умножения и деления.

6.3. Переборные методы

6.4. Поиск контуров и путей по матрице смежности

Наиболее простым способом идентификации путей и контуров являются матричные алгоритмы структурного анализа [107]. Они строятся на основе последовательного возведения в соответствующие степени матрицы смежности (см. 1.5).

Единица в матрице смежности **S** говорит о наличии пути между i-й и j-й вершинами длиной 1. Наличие 1 в (i, j)-й позиции в матрицы $\mathbf{S}^2$ означает путь длиной 2 между этими вершинами, и так далее. Таким образом, существование ненулевого значения на главной диагонали означает наличие пути из данной вершины в данную вершину, длинна которого равна степени матрицы. Значение матрицы смежности в различных степенях для графа, представленного на рис. 3.1 показаны ниже:

$$\mathbf{S}\quad \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix};\quad \mathbf{S}^2 \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix};\quad \mathbf{S}^3 \begin{bmatrix} 0 & 1 & 0 & 1 \\ 2 & 0 & 2 & 0 \\ 0 & 1 & 0 & 1 \\ 2 & 0 & 2 & 0 \end{bmatrix};\quad \mathbf{S}^4 \begin{bmatrix} 2 & 0 & 2 & 0 \\ 0 & 2 & 0 & 2 \\ 2 & 0 & 2 & 0 \\ 0 & 2 & 0 & 2 \end{bmatrix}.$$

Наличие 1 в главной диагонали $\mathbf{S}^2$ указывает на то, что четыре переменные системы входят в контуры длиной 2. Это позволяет определить вершины, входящие в контуры, его длину, но не конкретный вид. Поэтому требуется уточняющий переборный алгоритм на отобранных вершинах нелинейного системного гибридного графа, определяющего конкретный вид контура известной длины. На выходе этого алгоритма формируется дополняемый список из номеров вершин, входящих в каждый контур. С учетом различной длины контуров его удобнее представлять в памяти ПЭВМ динамическим списком

$$\mathbf{C} = \begin{bmatrix} 1 & 2, & & \\ 2 & 4, & & \\ 1 & 2 & 3 & 4, \end{bmatrix}.$$

Четвертая степень матрицы смежности $\mathbf{S}^4$ содержит информацию об еще одном контуре длиной 4. Но кроме этого повторяется информация о контурах длиной 2.

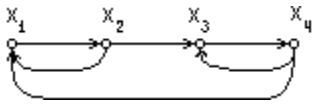


Рис. 3.1. Диаграмма графа одноуровневой модели СУ

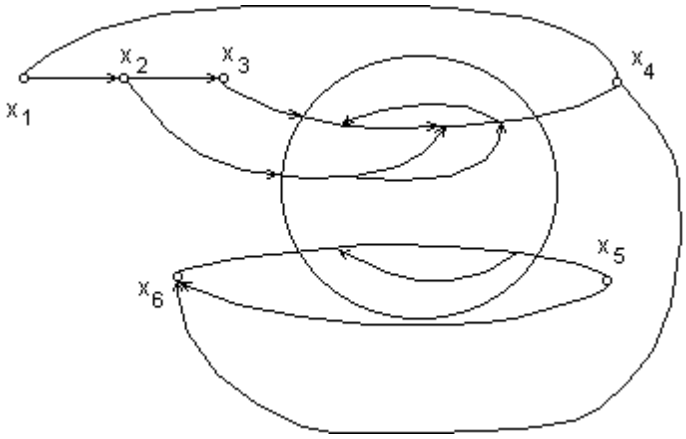


Рис. 3.2. Диаграмма графа иерархической модели СУ

Отмеченные особенности этого метода, повторение информации о контурах в матрицах более высокого порядка, кратного длине контура; трудности в обработки контуров одинаковой длины, требуют применения, в дополнению к рассматриваемому методу переборного алгоритма, уточняющего и отбрасывающего повторяющую информацию.

Наиболее существенным недостатком данного метода является его низкое быстродействие в следствие большого количества возведений матрицы смежности в соответствующие степени и большие затраты памяти ЭВМ для хранения информации.

6.5. Модифицированный алгоритм поиска контуров и путей по матрице смежности

Недостатки алгоритма поиска путей и контуров на основании представления топологии модели в форме матриц смежности, отмеченные выше, могут быть компенсированы, если использовать логические операции вместо математических и побитовое представление матрицы смежности. Быстрый рост необходимой памяти и временных затрат на работу алгоритма с ростом размерности систем в предлагаемом алгоритме компенсируются иерархическим представлением топологии модели а так же иерархическим характером построения алгоритмов топологического анализа.

Реализация алгоритма в этом случае использует не умножение, а логическую операцию И (в матрице присутствуют только значения 0 и 1), выполняемую одной машинной командой.

Как отмечалось ранее, составной характер представляемых моделей требует учета наличия связей между входами и выходами внутри подсистем. С этой целью водится матрица существования связи:

$$\mathbf{J_{ik}} = \begin{cases} 1, & \text{если внутри подсистемы существует связь между} \\ & \text{i входом и k выходом,} \\ 0, & \text{если такой связи нет.} \end{cases} \qquad (3.1)$$

Пример, иллюстрирующий данную особенность показан на рис. 3.2. При формировании матрицы смежности информация о внутренних контурах подсистемы не учитывается, учитывается только информация матрицы **J** (3.1) существования связи между входами и выходами подсистемы. Возведение в соответствующие степени матрицы смежности **S** позволяет выделить для данной системы 3 контура.

$$\underset{\mathbf{S}}{\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}}; \underset{\mathbf{S^2}}{\begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}; \underset{\mathbf{S^3}}{\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}}; \underset{\mathbf{S^4}}{\begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}.$$

Логическая сумма, - операции ИЛИ - позволяет определить все возможные связи между вершинами.

$$\mathbf{R} = \sum_{i=1}^{n} \mathbf{S}^{i} \qquad (3.2)$$

где n - размерность системы и, кроме того, определяет длину максимально возможного пути. Данную сумму называют матрицей достижимости.

Транспонированная матрица достижимости

$$\mathbf{Q} = \mathbf{R}^{\mathbf{T}}, \qquad (3.3)$$

называемой матрицей контрдостижимостей.

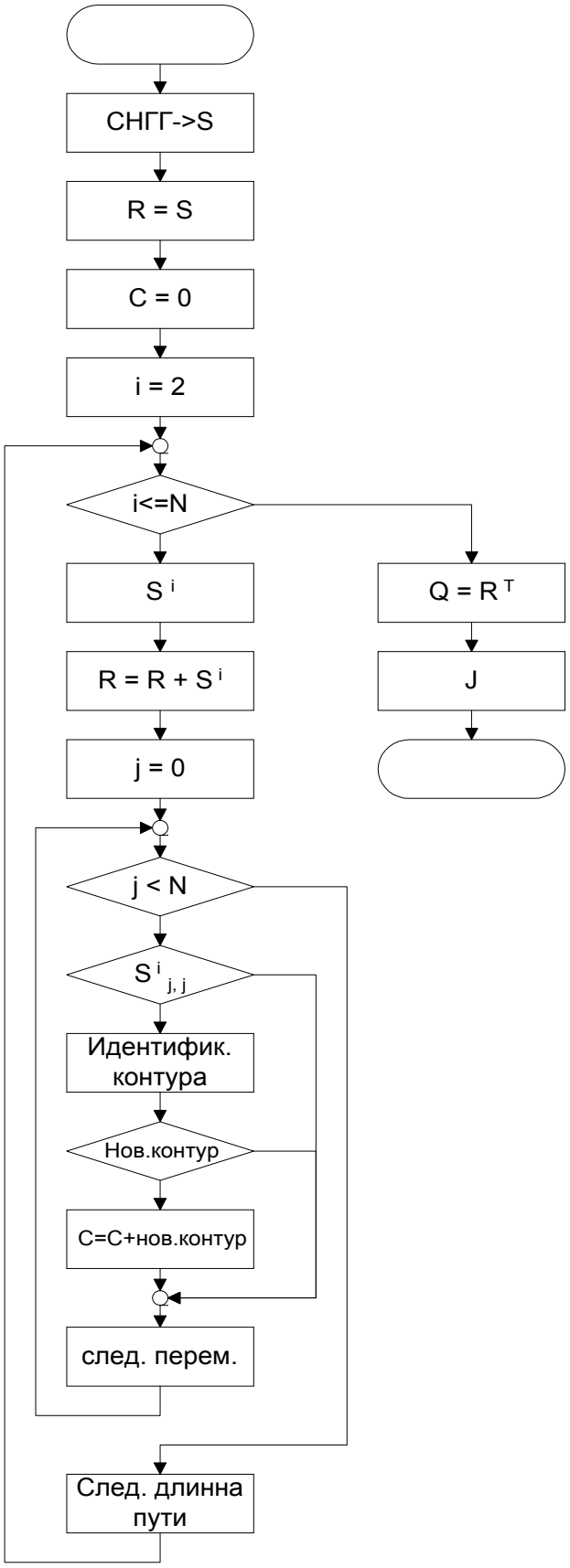
Ниже приведены значения матриц достижимости и контрдостижимости для системы представленной на рис. 3.2.

$$\mathbf{R} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \quad \mathbf{Q} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Если в матрице достижимости оставить только входные и выходные

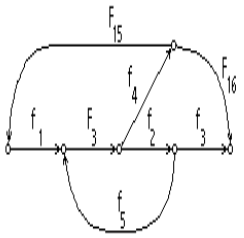
вершины подсистемы данного уровня иерархии, то получится матрица связей **J** (3.1), которая должна быть передана в вышестоящую по уровню систему для составления аналогичных функций структурного анализа и учета этой информации на стадии моделирования.

Блок схема, реализующая предлагаемый алгоритм, показана на рис. 3.4. Блок 1 выполняет преобразование из внутренней формы представления нелинейного системного гибридного графа в матрицу смежности размером *N* * *N*. Блоки 3 и 4 задают начальные значения матриц контуров **C** и достижимости **R**. Блоки 4,5,15 организуют основной цикл. Блок 6 вычисляет i-ю степень матрицы смежности, перемножение выполняется логической операцией “И”. Блок 7 выполняет накопление информации о всех возможных путях в матрице достижимости, суммирование производится логической операцией “ИЛИ”. Блоки 8,9,14 организуют цикл проверки вершин на принадлежность к контурам. В этом цикле перебираются элементы главной диагонали матрицы $\mathbf{S}^i$. Если вершина j относится к контуру, длиной i (блок 10), то в блоке 11 переборными методами этот контур выделяется. Выделенный контур, в блоке 12, сравнивается с уже существующими, хранящимися в списке контуров **C**. Если контур новый, то он добавляется к списку (блок 13). После завершения основного цикла, вычисляется матрица контрдостижимости **Q** (блок 16) и матрица связей в системе (подсистеме) **J** (блок 17), после чего алгоритм завершает свою работу. Выходными параметрами, возвращаемыми в вызвавшую программу, являются матрицы **R**,**Q**, **J**, **C**.



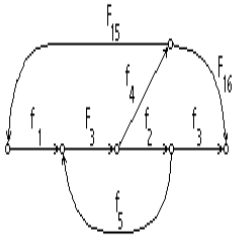
S=

0 1 0 0 0 0
0 0 1 0 0 0
0 0 0 1 1 0
0 1 0 0 0 1
1 0 0 0 0 1
0 0 0 0 0 0



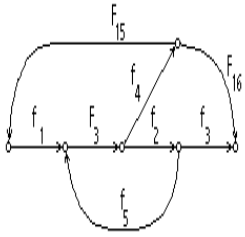
S^2 =

0 0 1 0 0 0
0 0 0 1 1 0
1 1 0 0 0 1
0 0 1 0 0 0
0 1 0 0 0 0
0 0 0 0 0 0



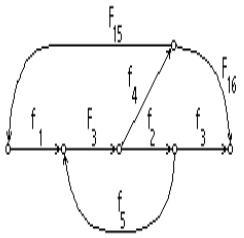
S^3 =

0 0 0 1 1 0
1 1 0 0 0 1
0 1 1 0 0 0
0 0 0 1 1 0
0 0 1 0 0 0
0 0 0 0 0 0



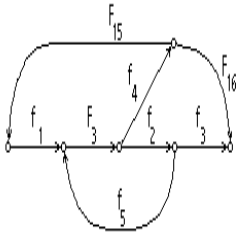
S^4 =

1 1 0 0 0 1
0 1 1 0 0 0
0 0 1 1 1 0
1 1 0 0 0 1
0 0 0 1 1 0
0 0 0 0 0 0



S^5 =

0 1 1 0 0 0
0 0 1 1 1 0
1 1 0 1 1 1
0 1 1 0 0 0
1 1 0 0 0 1
0 0 0 0 0 0



S^6 =

0 0 1 1 1 0
1 1 0 1 1 1
1 1 1 0 0 1
0 0 1 1 1 0
0 1 1 0 0 0
0 0 0 0 0 0

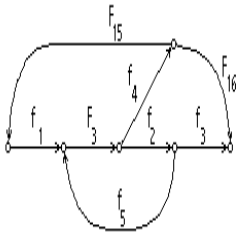
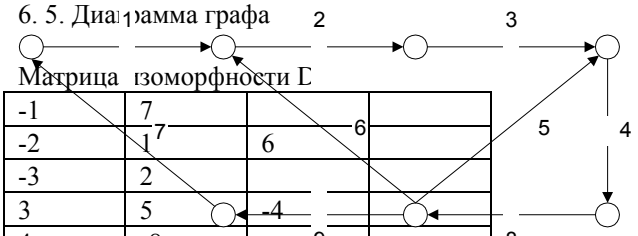


Рис. 3.4. Блок-схема и пример работы программы выделения контуров

6.6. Поиск контуров и путей по матрице изоморфности

6. 5. Диаграмма графа



Матрица изоморфности D

-1	7		
-2	1	6	
-3	2		
3	5	-4	
4	-8		
8	-5	-6	-9
9	-7		

Алгоритм идентификации контуров следующий:

1. Просмотреть строки матрицы. Для i-й строки просмотреть элементы до обнаружения отрицательного элемента $D_{i\,j} <0$. Запомнить номер строки и значение элемента $D_{i\,j}$.
2. Найти строки в матрице содержащие элемента $D_{k\,l} == - D_{i\,j}$. Для каждой найденной строки выполнить пп. 1, до тех пор пока в найденной последовательности повторно не вертится уже обнаруженная дуга, или программе не удастся обнаружить новую дугу, выходящую из этой вершины.

Пример для графа, представленного на рис. 6.5.

1. В 0-й строке нашел D[0][0]= -1 ;
2. Нашел D[1][1]= 1 ;
3. В 1-й строке нашел новый отрицательный элемент D[1][0] = -2 ;
4. Нашел D[2][1]= 2 ;
5. В 2-й строке нашел новый отрицательный элемент D[2][0] = -3 ;
6. Нашел D[3][0] = 3 ;
7. В 3-й строке нашел новый отрицательный элемент D[3][2] = -4 ;
8. Нашел D[4][0]= 4 ;
9. В 4-й строке нашел новые отрицательные элементы D[4][1] = -5, D[4][1] = -6, D[4][1] = -9. Необходимо разветвить поиск ;
10. Нашел D[3][1]= 5 ;
11. В 3-й строке нашел отрицательный элемент D[3][2] = -4. Этот элемент уже был в цепочки поиска. Поиск по этой ветки прекращен. Возвращаемся в пп.9 ;
12. Нашел D[1][2]= 6 ;
13. В 1-й строке нашел отрицательный элемент D[1][0] = -2. Этот элемент уже был в цепочки поиска. Поиск по этой ветки прекращен. Возвращаемся в пп.9 ;
14. Нашел D[6][0]= 9 ;
15. В 6-й строке нашел новый отрицательный элемент D[6][1] = -7 ;
16. Нашел D[0][1]= 7 ;
17. В 0-й строке нашел D[0][0]= -1. Этот элемент уже был в цепочки поиска. Поиск по этой ветки прекращен.
18. Новых отрицательных элементов в матрице не осталось поиск

прекращен.

После завершения поиска имеем:

-1	-2	-3	-4	-8		-5	-4	
						-6	-2	
						-9	-7	-1

По данным этого списка составляем контура.

Обычно в литературе указывается на высокую эффективность данного алгоритма. Он действительно выглядит очень просто и наглядно. Однако при его реализации придется организовывать разветвление работы алгоритма. Это можно сделать либо через рекурсию, либо организовать запоминание дерева перебора и незавершенные точки перебора. Что приводит к значительным затратам ресурсов ЭВМ, не учитывающихся большинством авторов анализирующих эффективность данного алгоритма.

6.6. Сравнение алгоритмов топологического анализа

К недостаткам представления топологии модели в форме матрицы смежности относят неэффективное использование памяти ЭВМ и низкое быстродействие алгоритмов. В качестве более эффективного способа в работе [107] предлагают матрицы изоморфности.

Неэффективность представления в памяти ЭВМ матрицы смежности обосновывается необходимостью помнить *N*N* элементов. Для матрицы изоморфности объем информации зависит от максимального числа дуг, входящих и выходящих из каждой вершины. Анализ большинства моделей СС НСУ показывает, что в среднем необходимо помнить *5*N* элементов (под средним значением здесь понимается среднее между матричной формой записи и среднем значением, получающимся при описании информации о топологии системы в форме динамического списка, с учетом выделения служебных полей этого списка для организации ссылок). Но в матрице смежности необходимо помнить только значения 0 или 1 и, соответственно, достаточно представить эту матрицу битовыми полями, в то время как для матрицы изоморфности понадобятся целые числа, занимающие в памяти ЭВМ 2 байта или 16 бит.

В результате для представления матрицы смежности будет необходимо *N*N/8* байт, а для матрицы изоморфности понадобится, (при максимальном числе входящих и выходящих дуг при вершине равным 5), - *10*N* байт. Сравнение объемов требуемой памяти показывает, что для описания топологий систем с размерностями меньшими, чем 80 переменных, эффективнее использовать матрицы смежности. При работе с системами большей размерности выгоднее использовать матрицы изморфности. Однако, учитывая рост затрат времени на расчет линеаризованной системы, получаемый в процессе решения по неявной схеме, работа с моделями больших размерностей не целесообразна. Предлагаемый подход ориентирован на составной и иерархический характер построения модели. Системы с большим числом независимых переменных в этом случае представляются комплексом подсистем, топология которых описывается в отдельных матрицах смежности.

По скорости работы алгоритма, поиск контуров по матрице изоморфности, приведенный в [107], близок к переборному алгоритму на графах [69]. Основные затраты времени в предлагаемом способе приходятся на возведение матрицы смежности в соответствующие степени. При выполнении этой операции с использованием операций умножения затраты времени будут значительными, однако, учитывая что в матрице присутствуют только значения 0 или 1, операцию умножения можно заменить логической операцией “И”, а сложение - логической операцией “ИЛИ”, которые выполняются значительно быстрее. Дополнительным способом повышения скорости является перемножение строки на столбец матрицы с использованием арифметической операции “И”. В этом случае за одну машинную команду “перемножаются” сразу по 16 элементов матрицы. Но для реализации этого понадобится хранить еще одну копию матрицы смежности, записанной по столбцам, а не по строкам, что при современном уровне развития средств вычислительной техники не является уже столь существенными затратами памяти компьютера. Из полученных векторов длиной 16 элементов, логической операцией “ИЛИ” получаем исходное значение.

Данный сравнительный анализ показывает, что представление топологии модели в форме матрицы смежности, по эффективности работы алгоритма поиска контуров и хранению в памяти не уступает представлению топологии в других формах представления.

Конкретные результаты сравнения эффективности алгоритмов по требуемой памяти и скорости работы во многом зависят от топологических особенностей рассматриваемого класса моделей, степени разряженности системы и особенностей программной реализации алгоритмов топологического анализа.

Можно предположить что эти характеристики имеют следующий вид:

Вставить вид характеристик.

6.7. Декомпозиция модели на топологическом ранге неопределенности

Традиционная декомпозиция модели основывается на выделении части графа в подсистему на основе принципа сильных связей, то есть связи элементов внутри подсистемы должны быть значительно сильнее, чем связи между ними и внешними элементами. Существенную часть этой работы, при иерархическом построении модели, выполняет сам пользователь, используя известную ему информацию о функциональном назначении подсистем исследуемого объекта.

В случае большеразмерной (крупномасштабной) системы, численное интегрирование неявными методами, как правило, не эффективно вследствие значительных временных затрат. Снизить затраты возможно в результате проведения редукции системы за счет формальной (искусственной) декомпозиции системы. Алгоритмы для такой декомпозиции на основе выделения сильных компонент можно найти в [63, 70, 107, 119, 120]. Однако в данной работе использован другой подход, связанный с ориентацией разрабатываемых алгоритмов на неявную схему моделирования. Предлагается

выделять последовательные цепи элементов или структуры без обратных связей, уравнения которых впоследствии интегрируются явными методами. Уравнения многовходовых элементов (нелинейные, линейные элементы суммирования и сравнения, суперблоки) и разветвления моделируются по неявной схеме.

В результате получается гиперграф, на ребрах которого образуются подсистемы, не содержащие обратные связи. Формируемая на основе преобразованного гиперграфа система уравнений моделируется по явной схеме интегрирования и периодически корректируется (балансируется) по выделенным переменным на основании неявной схемы.

Пример, иллюстрирующий данный подход, показан на рис. 3.5. Исходная система в виде графа (гиперграфа), представлена на рис. 3.5.а. Фрагменты структур, выделенные в процессе предлагаемой топологической декомпозиции для расчета по явной схеме, приведены на рис.3.5.б. Структура, предназначенная для расчета по неявной схеме представлена на рис.3.5.в. Информация, используемая при выполнении этой задачи, представляется в виде списка СНГГ и списка контуров **C**, получение которых рассматривалось в 3.2.

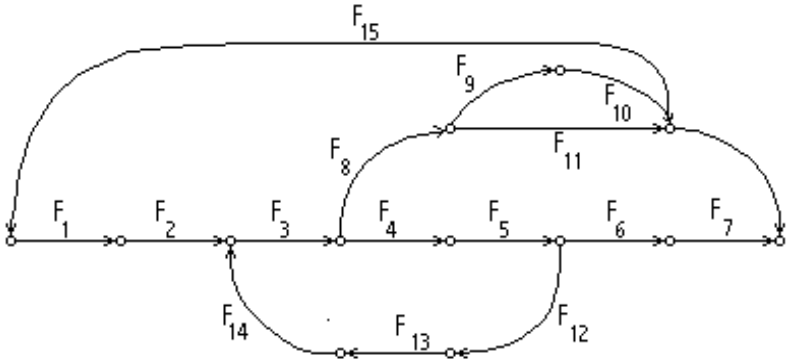


Рис. 3.5.а. Исходная структура модели

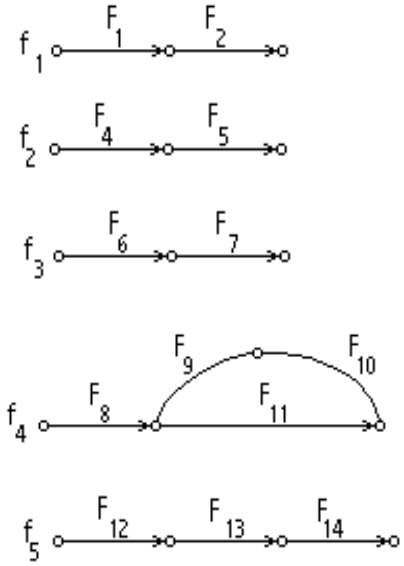


Рис. 3.5.б. Подсистемы, выделенные по предлагаемой методике, для

которых используется явная схема

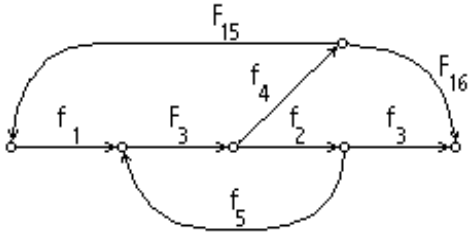


Рис. 3.5.в. Структура, оставленная по предлагаемой методике для расчета по неявной схеме

Эффект применения такого метода декомпозиции связан с увеличением скорости расчетов. Это вызвано снижением размерности подсистемы данного уровня иерархии, рассчитываемого неявными методами. Моделирование выделенных подсистемы не несет значительных вычислительных затрат, вследствие того, что подсистемы рассчитываются явными методами. И если проанализировать график затрат времени расчета по полностью неявной схеме, то при использовании декомпозиции системы, представленной на рис. 2.5, выигрыш по затратам времени составляет более 50 %.

В общем случае расчета всех подсистем по неявной схеме эффект от декомпозиции можно оценить как:

$$K = \frac{N^3}{\sum_{i=1}^{n} N_i^3},$$

где N - размерность исходной системы, N_i - размерность i подсистемы, n - количество подсистем.

В случае моделирования выделенных подсистем по явной схеме расчета вычислительный эффект, связанный с повышением скорости вычислений можно оценить как:

$$K = \frac{k \cdot N^3}{k \cdot N_{неявн.}^3 + \sum T_{явн}^i},$$

где k - коэффициент быстродействия вычислений по неявной схеме, $T_{явн}^i$ - время моделирования i-ой выделенной подсистемы данного уровня по явной схеме расчета. Причем из сравнения скоростей расчета по явным и неявным схемам известно что $k \cdot N_{неявн}^3 >> \sum T_{явн}^i$.

Переборный алгоритм, реализующий подобную декомпозицию, приведен на рис. 3.6. В блоках 1 и 2 обнуляются исходные списки элементов, вычисляемых по явной и неявной схеме соответственно. В блоке 3 происходит перенумерация номеров переменных. Критерий для сортировки номеров переменных устанавливается так, чтобы переменная на входе блока имела номер меньше, чем на выходе. Нарушение этого порядка означает наличие

обратной связи.

В основном цикле по всем переменным рассматриваемой модели (блоки 4, 5 и 10), производится их разделение на два динамических списка.

В том случае, если относительно рассматриваемой переменной (вершины) обнаруживается замкнутый контур, (номер выходной переменой меньше чем номер входной переменной, или одной из входных переменных (блок 6)), то соответствующее уравнение присоединяется к части модели, рассчитываемой по неявной схеме (блок 8). В блоке 7, в случае разветвления, когда выходная переменная, являющаяся следствием этого уравнения присутствует в качестве причины сразу в нескольких уравнениях, производится дополнительный анализ на предмет необходимости рассчитывать и это уравнение по неявной схеме. Если это разветвление сводится к соединению эквивалентному последовательной цепи элементов, например элемент в приведенном на рис. 3.5 выделенном фрагменте (обозначенным как - f_4), соответствующие это уравнение не требуется рассчитывать по неявной схеме, и оно “отправляется” в другой список (блок 9).

6.8. Сортировка модели на топологическом ранге неопределенности

Применяемые различные математические методы, переключения между ними в процессе моделирования, требуют различных подходов к упорядочиванию элементов в подсистемах модели, иначе говоря, записи уравнений. При использовании явных методов традиционно первыми решают алгебраические уравнения, то есть фактически происходит упорядочивание системы уравнений по этому признаку. Предлагаемые адаптивные алгоритмы требуют построения списков на основании причинно-следственных взаимоотношений. Неявные методы, строго говоря, не предполагают упорядочивания элементов, но для повышения скорости расчетов целесообразно провести определенную сортировку. Порядок следования уравнений для всех этих методов различен.

Наиболее сложными и трудоемкими являются неявные методы. Поэтому целесообразно в качестве основного порядка следования уравнений в подсистемах принять порядок элементов, используемый для неявных методов. Порядок следования уравнений для остальных методов записывается в индексные массивы. При переключении с одного алгоритма на другой новые перестановки не производятся и порядок расположения элементов берется из соответствующего массива индексов.

Основные потери быстродействия при численном интегрировании по неявной схеме возникают при решении линейной системы уравнений [85]. Для ускорения этого процесса предлагается на топологическом уровне представления модели расположить уравнения (номера элементов) так, чтобы ненулевые элементы в матрице Якоби (3.9) были расположены в заранее определенном порядке следования. Такое упорядочивание элементов позволяет использовать быстрые, специализированные алгоритмы решения получаемых на каждой итерации систем линейных уравнений [72, 86, 106].

Предлагаемый алгоритм [А14, А17, А24, А35, А44], схема которого представлена на рис. 3.7, предполагает приведение системы к форме, при которой образовывается ленточная матрица (рис. 3.8). Широкий класс алгоритмов для работы с подобными матрицами представлен в работах [1, 72, 86, 92, 96, 106, 108, 122]. Кроме того, необходимо отметить, что приведение к матрице специального вида происходит на уровне топологических моделей, а не на вычислительной стадии расчета.

На рис. 3.9-3.12 представлены результаты проведения предложенной сортировки для тестовой моделей и модели ТГУ-532, показаны матрицы смежности исходных и преобразованных моделей.

Алгоритм построен на основе традиционных обменных методов сортировок [57]. В блоке 1 формируется булевская матрица ненулевых значений матрицы Якоби. Ненулевые элементы в матрице Якоби образуются за счет переменных, являющихся причиной и переменных, являющихся следствием каждого уравнения. Очевидно что это можно записать в матричной форме как:

$$\mathbf{A} = \mathbf{C}^{\mathrm{T}} + \mathbf{I},$$

где **I** - единичная матица, **C** - матрица смежности, т - символ транспонирования, **A** - матрица наличия ненулевых значений матрицы Якоби. Учитывая, что большинство элементов системы составляют элементы типа SISO (один вход и один выход), то матрица будет сильно разряжена.

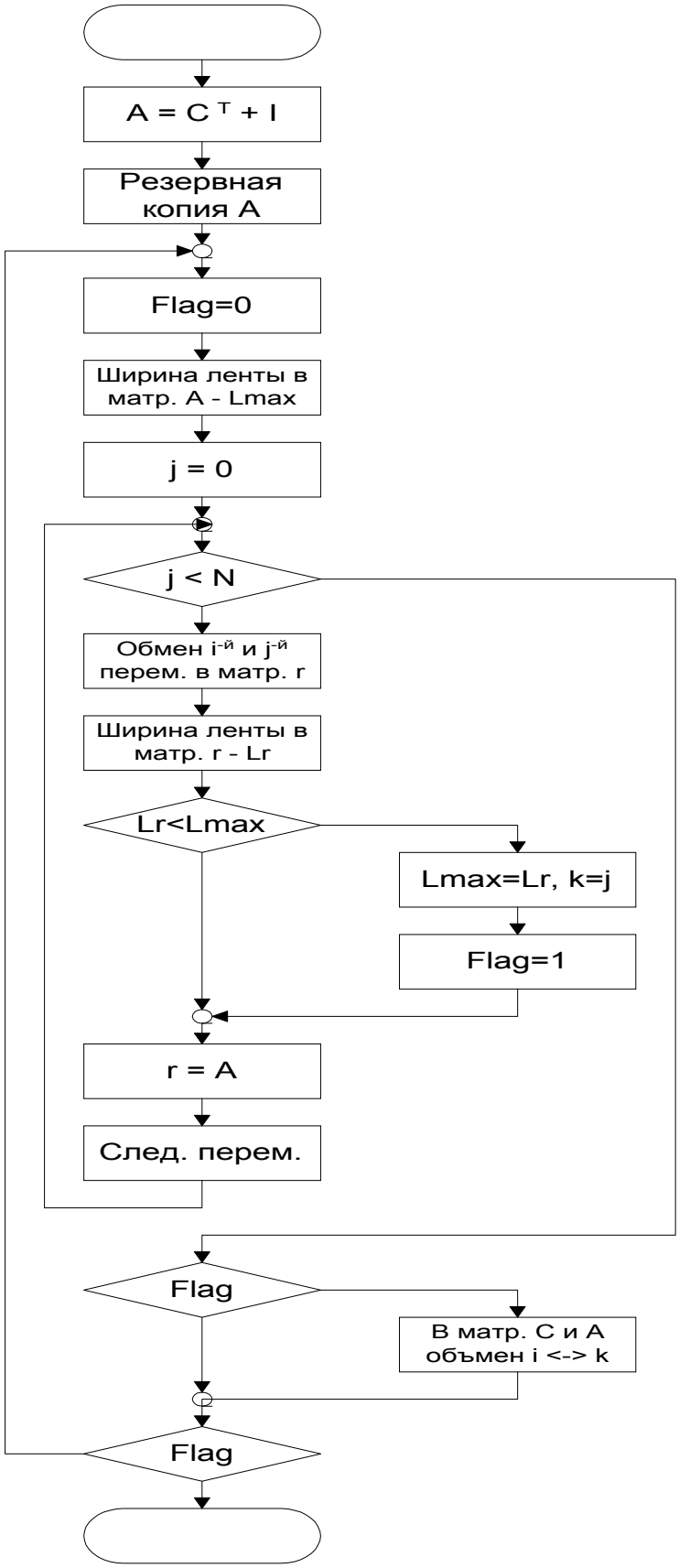


Рис. 3.7. Блок схема алгоритма сортировки на топологических моделях

В блоке 2 создается копия матрицы **A**, на которой производятся перестановки (**r**). В блоках 3, 5 организуют основной цикл по переменной *flag*,

устанавливаемый в блоке 4 в нуль. В матрице **A**, в блоке 5, определяется максимальное расстояние от ненулевого элемента до единичной диагонали *lmax*, рис. 2.8, и его номер *i*. Блоки 6,7,14 организуют цикл, где *N* размерность системы. Блок 9 производит обмен в матице **r** *i* и *j* элементов. При этом, в матрице **r**, определяется максимальное расстояние до ненулевого элемента (блок 9). Если оно больше определенного *lmax*, то присваивается новое значение *lmax* и запоминается при какой перестановке строк оно было достигнуто. Переменная *flag* устанавливается в 1 (блоки 10,11,12). В блоке 13 возвращается исходное значение матрице **A**. После завершения цикла (6,7,14) проверяется значение переменной *flag*. Если *flag* == 1 (истина) то производятся перестановки в СНГГ, и в соответствующих ему матричных формах представлений **C**,**A** (блоки 15,16). Если была произведена перестановка, то работа алгоритма возвращается на п. 4. Если перестановка не была произведена, то получено минимальное значение *lmax* и алгоритм завершает свою работу.

Подобные перестановки для упрощения расчетной формы модели были предложены Д. Стюардом, а также рассмотрены в [119]. Предлагаемый в них алгоритм предполагает приведение вида матрицы к блочно треугольному виду. Это упрощает расчеты, но не позволяет без потери информации перейти на более быстрые методы, так как матрица остается почти треугольной, а не треугольной [119]. Кроме того, формальный принцип образования диагональных блоков и отказ от учета влияния “отсоединенных частей”, может привести к потере существенной информации о поведении модели.

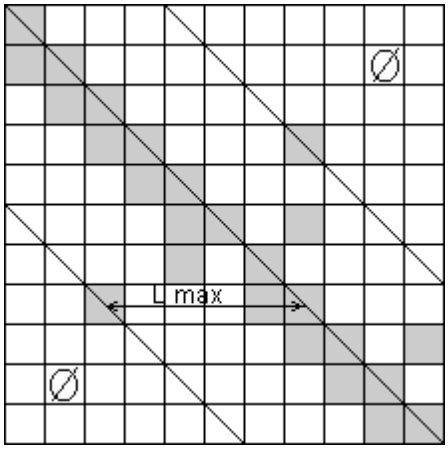
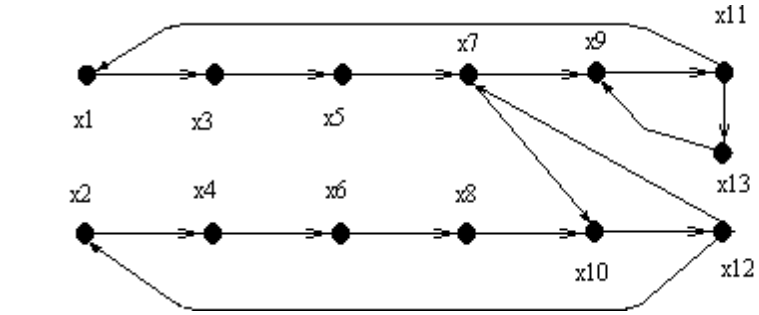


Рис. 3.8. Ленточная матрица



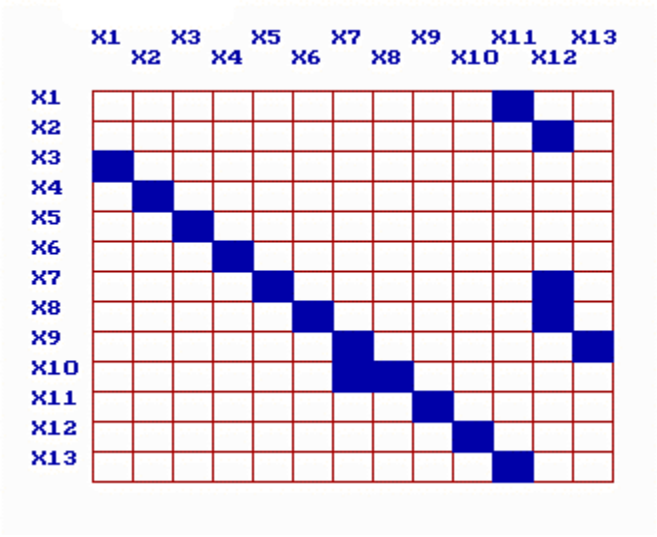
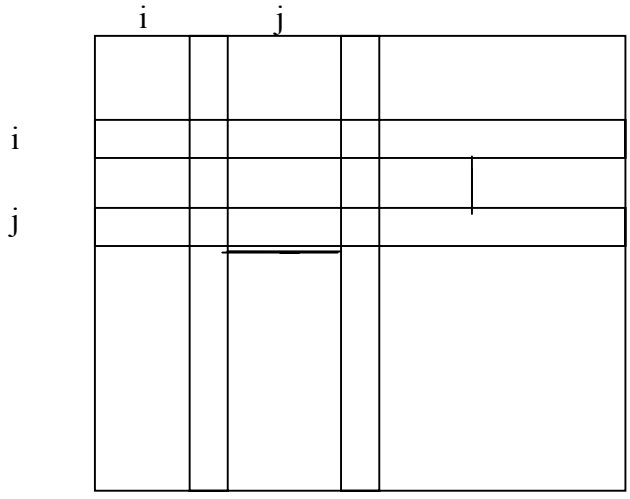


Рис. 3.9. Тестовая модель и ее исходная матрица смежности



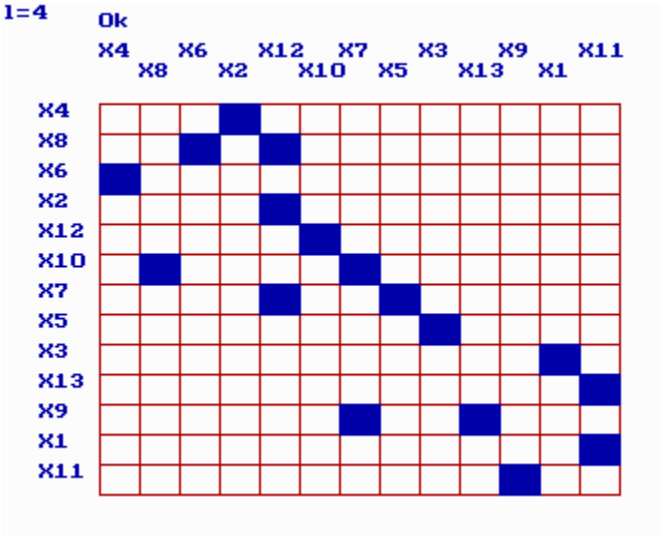


Рис. 3.10. Иллюстрация сортировки и ленточная матрица отсортированной системы

Пример, показанный на рис. 3.10, иллюстрирует работу этого алгоритма. В отличии от приведенного в [119], он позволяет использовать для решения линеаризованной системы ленточные методы, за счет чего можно достигнуть увеличения скорости расчета. Используя оценки скорости вычислений для ленточных матриц, приведенные в [86] и проверенные рядом экспериментов, эффект от применения этакого подхода можно оценить как

$$K = \frac{3}{2} * \frac{N}{M},$$

где N - размерность системы, а $2*M+1$ - ширина ленты (на рис. 3.8 $M=lmax$).

(Для примера, приведенного на рис. 3.10, в результате предлагаемой сортировки, представлены на рис. 3.11. Улучшения по сравнению с традиционными методами составят)

Для тестовой модели, которая представлена на рис. 3.9 в виде графа и матрицы смежности, предлагаемая сортировка порядка уравнений (номеров переменных) приводит матрицу к виду, показанному на рис. 3.10. Данное преобразование позволяет получить вычислительный эффект, связанный с увеличением скорости расчетов, в

$$K = \frac{3}{2} * \frac{13}{4} = 5 \text{ раз.}$$

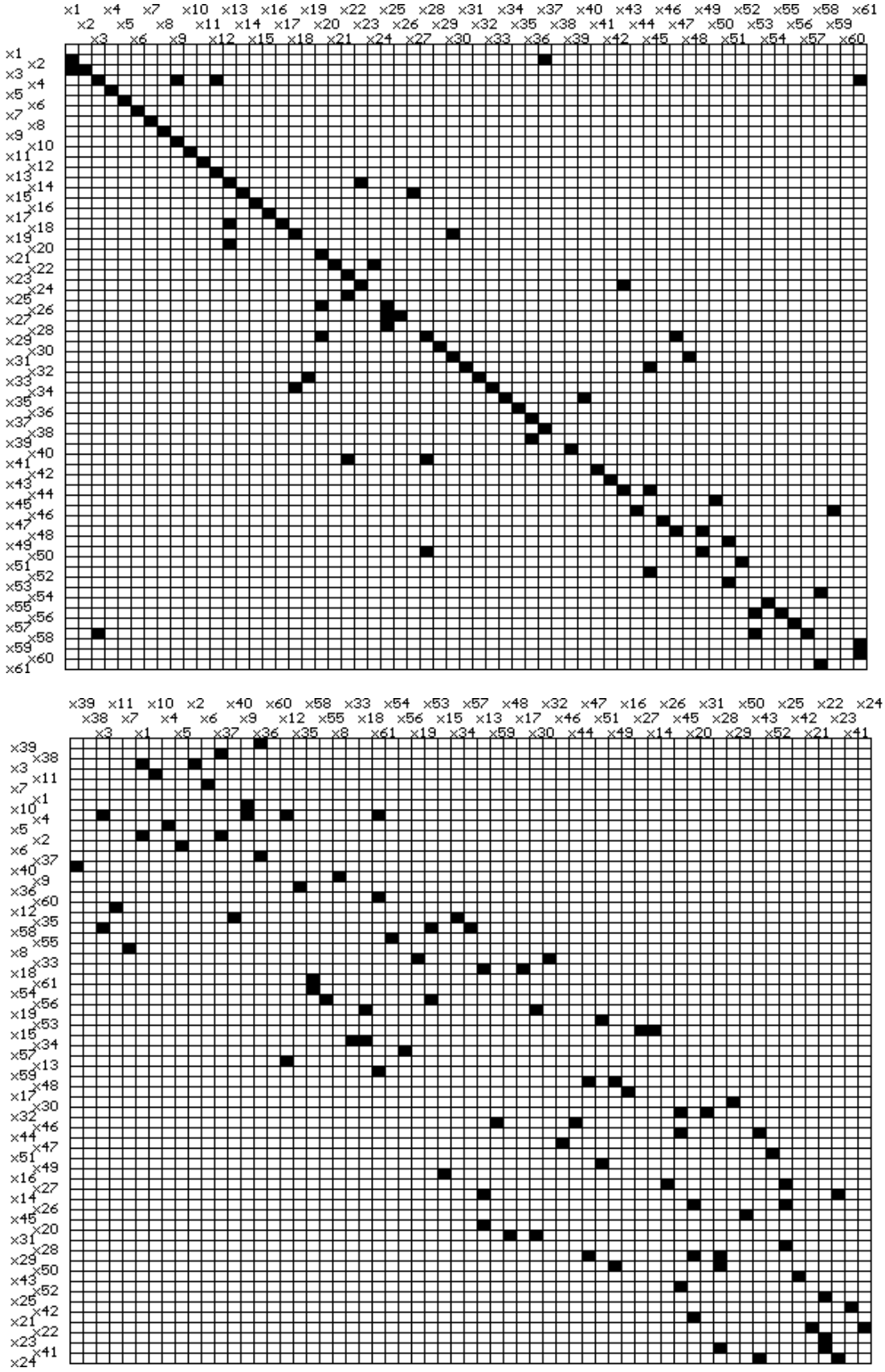


Рис. 3.12. Матрица смежности исходной и отсортированной модели

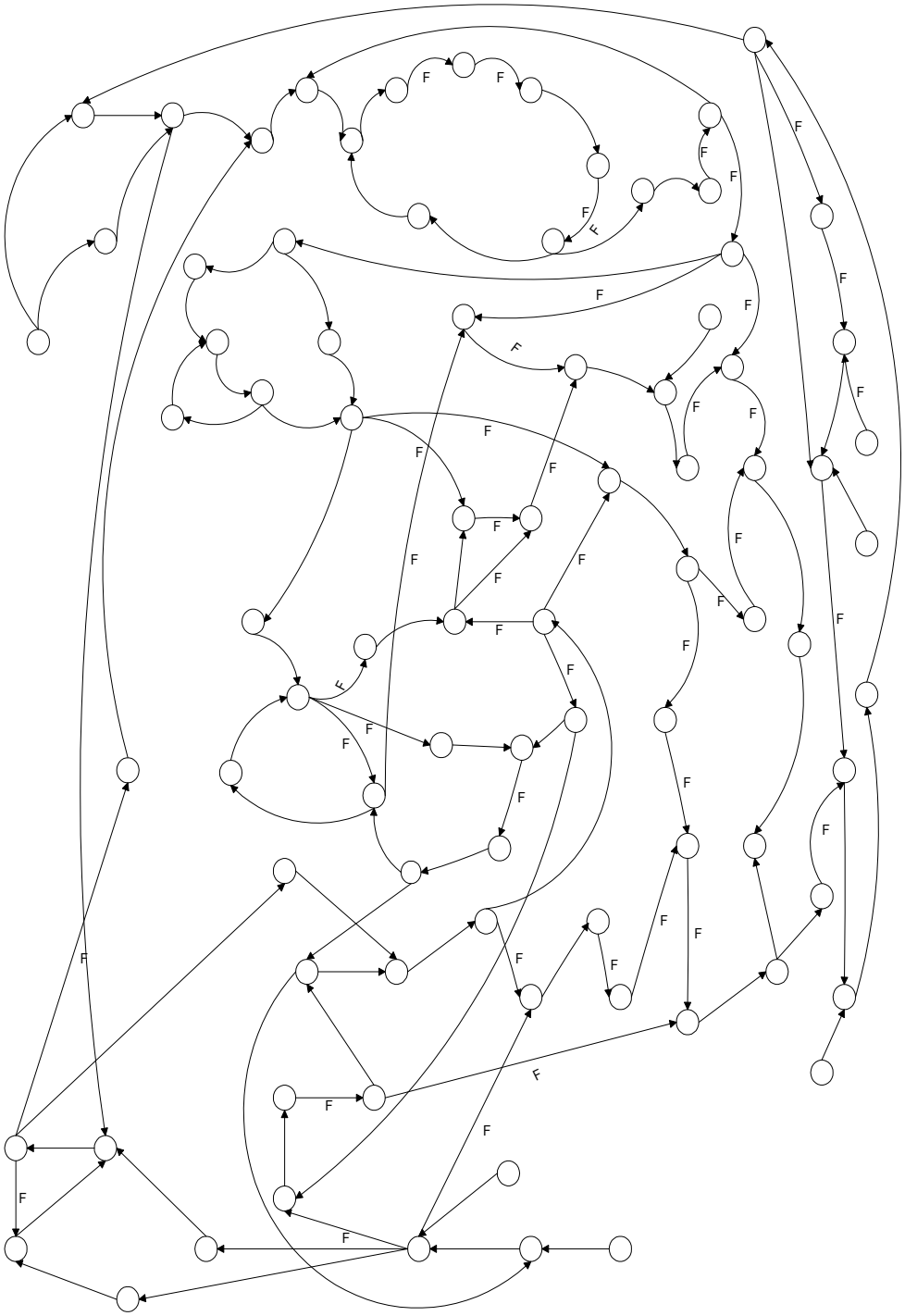


Рис. в.1 Диаграмма графа модели структурно-сложной нелинейной системы управления турбоагрегата электростанции

Еще лучше достоинства этого подхода видны на примере системы представленной на рис. в.1. Результаты сортировки показаны на рис. 3.12. Эффект от применения предлагаемой сортировки составил:

$$K = \frac{3}{2} * \frac{61}{16} = 5.7 \text{ раз.}$$

Предложенный алгоритм сортировки элементов приводит к получению матрицы Якоби известного вида, что позволяет использовать более быстрые алгоритмы решения систем нелинейных уравнений в процессе моделирования по неявной схеме. Данный подход отличается от встречаемых в литературе тем, что учитываются все переменные, без исключения части из них, соответствующих слабым связям. Кроме того, приведение к матрице специального вида происходит на уровне топологических моделей, а не на вычислительной стадии расчета. Эффект от применения такой сортировки, выполняемой однократно, получается на каждой итерации расчета для каждого момента времени.

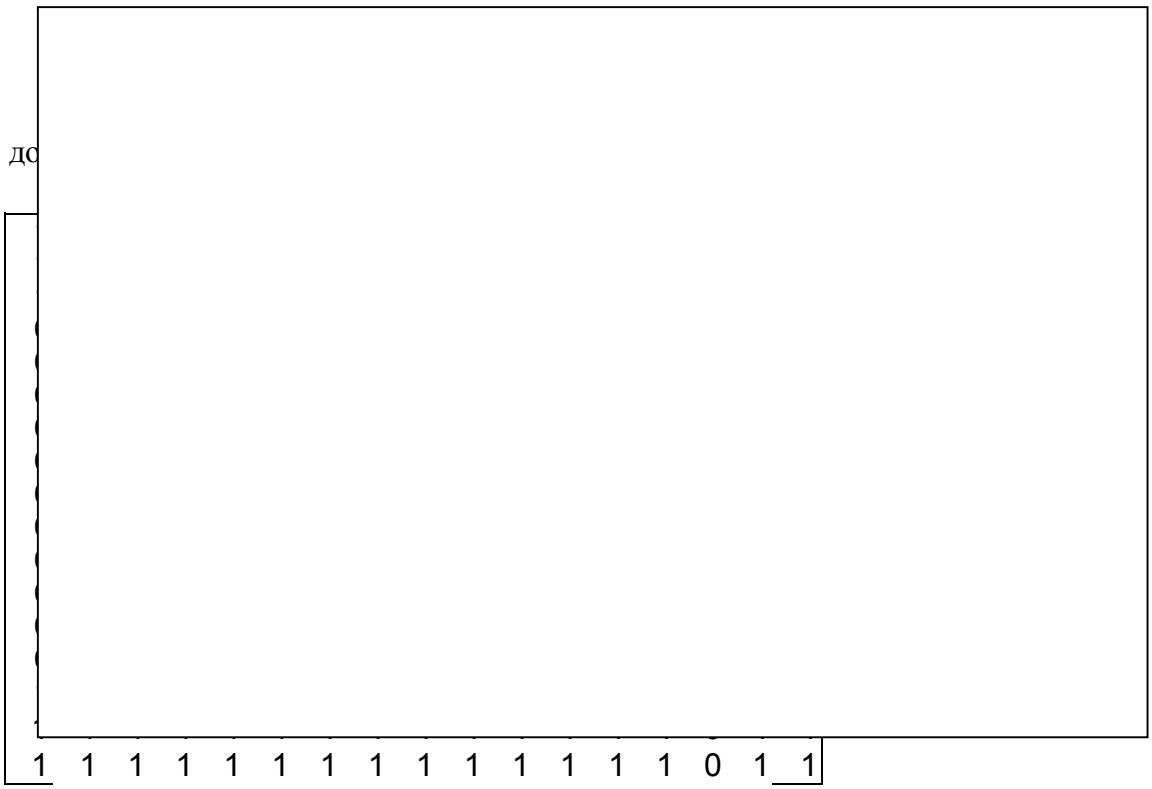
6.9. Нахождение сильных компонент графа

Нахождение сильных компонентов графа широко используется в процессе декомпозиции исходной модели на подсистемы, приведение множества уравнений к блочному виду. Возможно, привести пример – размещение компонентов принципиальной схемы на плате и многое другое.

Наиболее простым является следующий алгоритм:

Для системы представленной на рис. 6.9, строим матрицу пересечений.

В матрице пересечений W, w_{ij} =1, если есть путь и из i-й вершины в j-ю, и обратно.



до

1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1

Матрица контрдостижимости Q

1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1

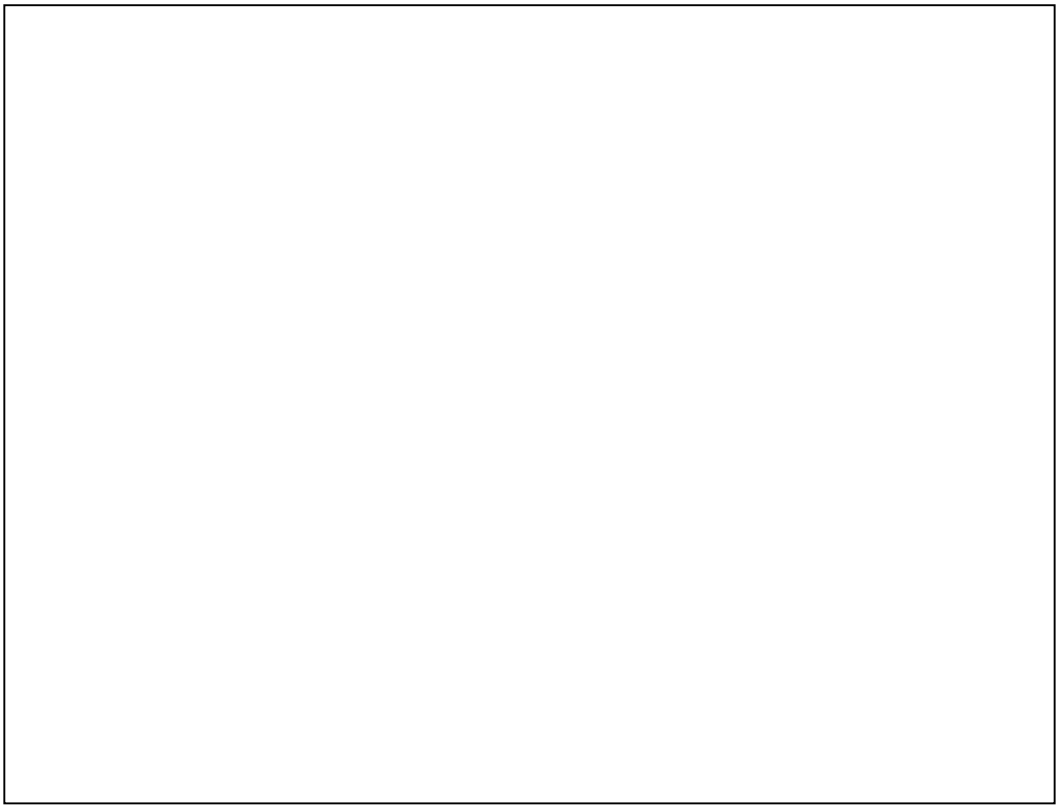
1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1
1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1
1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1
1	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1
1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1
1	1	1	1	1	1	1	0	0	0	0	0	0	0	1	1	1
1	1	1	1	1	1	1	1	1	1	1	0	0	0	1	1	1
1	1	1	1	1	1	1	1	1	1	1	0	0	0	1	1	1
1	1	1	1	1	1	1	1	1	1	1	0	0	0	1	1	1
1	1	1	1	1	1	1	1	1	1	1	0	0	0	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1
1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Для системы, представленной на рис. 6.9, матрица пересечений имеет вид.

1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1	1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1	1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1
1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1	1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1	1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1
1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1	1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1	1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1
0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0	0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0	0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0
0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0	0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0	0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0
0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0	0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0	0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0
0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0	0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0	0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0	0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0	0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0	0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0	0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0	0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0	0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1	1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1	1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1
1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1	1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1	1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1

Матрицу пересечений можно получить как P= R И Q.

В матрице пересечений выбираются блоки элементов с симметричным расположением 1 в строках и столбцах.



Заключение

Алгоритмы топологического анализа имеют очень важное значение, многие задачи исследования систем могут быть решены на топологическом уровне. Повышение эффективности всех стадий исследования системы возможно в первую очередь за счет учета топологических особенностей модели.

Оглавление