МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ
Кафедра компьютерных технологий и систем

Березко Иван Сергеевич

РЕАЛИЗАЦИЯ МЕТОДОВ ИДЕНТИФИКАЦИИ И УПРАВЛЕНИЯ СЕНСОРНОЙ СЕТКОЙ: АЛГОРИТМЫ, ТЕХНОЛОГИИ, ПРОГРАММНЫЕ СРЕДСТВА И ПРИЛОЖЕНИЯ

Дипломная работа

Научный руководитель:
кандидат физ.-мат. наук, доцент
Л. А. Пилипчук

Допущена к защите
«___» _____ 20__ г.
Заведующий кафедрой компьютерных технологий и систем
доктор педагогических наук, профессор В. В. Казаченок

Минск, 2025

ОГЛАВЛЕНИЕ

РЕФЕРАТ

Структура и объем дипломной работы

65 страницы, 13 рисунков, 2 таблицы, 3 приложения, 16 источников

Ключеные слова: СИСТЕМА ЛИНЕЙНЫХ УРАВНЕНИЙ, СЕНСОР, СЕТЬ, ДУГОВОЙ ПОТОК, ГРАФ, ДУГА, ВЕРШИНА, МАТРИЦА ИНЦИДЕНТНОСТИ, ВИЗУАЛИЗАЦИЯ, ОПТИМАЛЬНОЕ РЕШЕНИЕ, СУБОПТИМАЛЬНОЕ РЕШЕНИЕ, МНОЖЕСТВО

Текст реферата

Объект исследования - сеть различной размерности и конфигурации, система алгебраических уравнений с соответствующей матрицей инцидентности графа.

Цель исследования - задача поиска различных конфигураций сенсоров сети для оценки потоков, решение системы линейных уравнений на основе собранной после установки сенсоров информации.

Методы исследования - чтение документации используемых программных средств, изучение соответствующей литературы, изучение методов решения задачи расположения сенсоров для оценки потока.

Полученные результаты—программная реализация генерации связного графа и сохранение сгенерированных данных в файл, программная реализация субоптимального решения, позволяющая получить решение в символьном виде на основе информации, полученной после установки определенной конфигурации сенсоров, поэтапная визуализация полученных результатов

Область возможного применения—сферы, охватывающие задачи оптимизации и мониторинга в различных инженерных системах, анализ потоков в транспортных сетях и логистике, теория графов, решение систем линейных уравнений, непосредственно связанных с теорией матриц, в частности с матрицами инцидентности графа.

РЭФЕРАТ

Структура і аб'ём дыпломнай работы

65 старонкі, 13 малюнкаў, 2 табліцы, 3 дадаткі, 16 крыніц

Ключавыя словы: СІСТЭМА ЛІНЕЙНЫХ РАЎНЯННЯЎ, СЕНСАР, СЕТКА, ДУГОВЫ ПОТАК, ГРАФ, ДУГА, ВЕРШЫНА, МАТЫЦА ІНЦЫДЭНТНАСЦІ, ВІЗУАЛІЗАЦЫЯ, АПТЫМАЛЬНАЕ РАШЭННЕ, СУБОПТЫМАЛЬНАЕ РАШЭННЕ, МНОЖАСТВА

Змест прафы

Аб'ект даследавання - сетка рознай памернасці і канфігурацыі, сістэма алгебраічных раўнянняў з адпаведнай матрыцай інцыдэнтнасці графа.

Мэта даследавання - задача пошуку розных канфігурацый сенсараў сеткі для ацэнкі патокаў, рашэнне сістэмы лінейных раўнянняў на аснове сабранай пасля ўсталёўкі сенсараў інфармацыі.

Метады даследавання - чытанне дакументацыі выкарыстаных праграмных сродкаў, вывучэнне адпаведнай літаратуры, вывучэнне метадаў рашэння задачы размяшчэння сенсараў для ацэнкі патоку.

Атрыманыя вынікі – праграмная рэалізацыя генерацыі злучальнага графа і захаванне згенераваных дадзеных у файл, праграмная рэалізацыя субоптымальнага рашэння, якая дазваляе атрымаць рашэнне ў сімвалічным выглядзе на аснове інфармацыі, атрыманага пасля ўсталявання пэўнай канфігурацыі сенсараў, паэтапная візуалізацыя атрыманы вынікаў.

Вобласць магчымага практычнага прымянення—сферы, якія ахопліваюць задачы аптымізацыі і маніторынгу ў розных інжынерных сістэмах, аналіз патокаў у транспартных сетках і лагістыцы, тэорыя графаў, рашэнне сістэм лінейных раўнянняў, непасрэдна звязаных з тэорыяй матрыц, у прыватнасці з матрыцамі інцыдэнтнасці графа.

SUMMARY

Structure and scope of the diploma work

65 pages, 13 figures, 2 tables, 3 appendices, 16 sources

Keywords: SYSTEM OF LINEAR EQUATIONS, SENSOR, NETWORK, ARCS FLOW, GRAPH, ARC, VERTEX, INCIDENCE MATRIX, VISUALIZATION, SET, OPTIMAL SOLUTION, SUBOPTIMAL SOLUTION

Summary Text

The object of Research - a network of varying size and configuration, a system of algebraic equations with the corresponding incidence matrix of the graph.

The aim of the research - the task of finding different configurations of network sensors to evaluate flows, solving the system of linear equations based on the information collected after the installation of sensors.

Research Methods - reading documentation of the used software tools, studying relevant literature, exploring methods for solving the sensor placement problem to estimate the flow.

The results of the work — software implementation of generating a connected graph and saving the generated data to a file, software implementation of a suboptimal solution, which allows obtaining a solution in symbolic form based on information received after the installation of a specific sensor configuration, step-by-step visualization of the obtained results.

Recommendations on the usage — areas covering optimization tasks and monitoring in various engineering systems, flow analysis in transportation networks and logistics, graph theory, solving systems of linear equations, directly related to matrix theory, specifically to incidence matrices of graphs.

ВВЕДЕНИЕ

В условиях стремительного роста сложности и масштабов сетевых инфраструктур - от транспортных и информационных систем до телекоммуникационных и энергетических сетей - эффективное управление и контроль потоками становятся критически важными составляющими. С ростом объема передаваемой информации и усложнением топологии таких сетей возрастает необходимость в применении интеллектуальных средств наблюдения, включая программируемые сенсоры и устройства сбора данных.

Современные технологии передачи и обработки информации позволяют собирать данные о потоковых характеристиках даже в очень больших сетях, не охватывая каждый узел напрямую. Как отмечается в [7], достаточно установить необходимые устройства наблюдения лишь в ограниченное количество участков сети, чтобы она была полностью просматриваемой. Среди всевозможных стратегий расположения устройств в узлах сети, особую важность представляет разработка оптимальной стратегии - стратегии, предполагающей использование минимального числа сенсоров и позволяющей осуществлять сбор информации о потоковых характеристиках сети.

Наиболее интуитивной является стратегия полного перебора всевозможных вариантов расстановки сенсоров. Однако, как описано в [7], задача нахождения минимального набора сенсоров, обеспечивающих полное наблюдение за потоками в сети, относится к классу NP-полных. С увеличением масштабов сети методы полного перебора для определения оптимальных конфигураций обозреваемых узлов оказываются неэффективными и неприемлемыми в связи со значительными временными затратами и высокой вычислительной сложностью. Исходя из этого, можно сделать вывод, что основной задачей является разработка новых эффективных алгоритмов идентификации конфигурации сенсоров. В [7], [15] представлены эффективные алгоритмы и технологии численного решения исследуемой задачи. Однако данные алгоритмы не являются оптимальными, что еще раз подчеркивает сложность данной данной задачи.

ГЛАВА 1. ОПТИМАЛЬНОЕ РЕШЕНИЕ ЗАДАЧИ РАСПОЛОЖЕНИЯ СЕНСОРОВ ДЛЯ ОЦЕНКИ ПОТОКОВ НА НЕНАБЛЮДАЕМОЙ ЧАСТИ СЕТИ

1.1 Общие теоретические сведения

В качестве математической модели задачи идентификации потоковых характеристик в сети рассматривается система линейных алгебраических уравнений, в которой число уравнений меньше числа неизвестных [14]. Подобная система называется недоопределенной и имеет вид

$$\sum_{j \in I_i^+(U)} x_{i,j} - \sum_{j \in I_i^-(U)} x_{j,i} = \begin{cases} x_i, i \in I^* \\ 0, i \in I \backslash I^* \end{cases} \tag{1.1}$$

где $G = (I, U)$ - связный ориентированный двунаправленный граф с множеством узлов I и множеством дуг U, $I^* \subseteq I, I^* \neq \varnothing$. Множество вершин I определенно на $I \times I$, оба множества вершин I и дуг U являются конечными: $|I| < \infty, |U| < \infty$. $I_i^+(U) = \{j \in I : (i,j) \in U\}$, $I_i^-(U) = \{j \in I : (j,i) \in U\}$. В рассматриваемом двунаправленном графе G наличие ориентированного ребра $(i,j) \in U$ с соответствующим дуговым потоком $x_{i,j}$ предполагает существование противоположного ребра $(j,i) \in U$, для которого дуговой поток $x_{j,i}$, в общем случае, отличен от $x_{i,j}$.

Введем множество M, содержащее узлы графа G с расположенными в них устройствами для сбора информации, $M \subseteq I$. Задача размещения сенсоров сводится к определению такого подмножества узлов M в графе $G = (I, U)$, моделирующем транспортную или иную потоковую сеть, при котором количество узлов с установленными измерительными устройствами (сенсорами) минимально, но конфигурация этих сенсоров обеспечивает возможность наблюдения за потоками в системе. Предполагается, что в каждом узле из множества M установлены средства сбора данных, благодаря чему доступна априорная информация в виде показаний сенсоров. Эту информацию можно описать при помощи следующих равенств [13]:

$$\begin{gathered} x_{i,j} = f_{i,j}, j \in I_i^+(U), x_{j,i} = f_{j,i}, j \in I_i^-(U), i \in M, \\ x_i = f_i, i \in M \bigcap I^*. \end{gathered} \tag{1.2}$$

Здесь значения потоков по дугам $f_{i,j}$ и значения внешних потоков f_i получены заранее из соотношений (1.2) при использовании данных, собранных сенсорами. Далее используется дополнительная информация о долях разбиения трафика. Пусть известны параметры $p_{i,j}$ именуемые коэффицентом разбиения потока, где $0 < p_{i,j} \leq 1$, $(i,j) \in U$. Тогда общий выходной поток из узла i, обозначаемый $F(i) = \sum_{j \in I_i^+(U)} x_{ij}$, можно выразить как сумму компонентных величин.

Поскольку коэффициент $p_{i,j}$ определяют, какая часть общего потока направляется по дуге (i,j), при условии, что $|I_i^+(U)| \geq 2$, для произвольной исходящей дуги (i,v_i) можно представить остальные потоки, выходящие из узла i, в виде системы [13]:

$$x_{ij} = \frac{p_{i,j}}{p_{i,v_i}} x_{i,v_i}, j \in I_i^+(U) \backslash \{v_i\}, \left|I_i^+(U)\right| \geq 2, \tag{1.3}$$

где (i,v_i) - каноническая дуга, исходящая из узла. Пусть $\beta_{i,j} = \frac{p_{i,j}}{p_{i,v_i}}$, если $\left|I_i^+(U)\right| \geq 2$, и $\beta_{i,j} = p_{i,v_i}, \left|I_i^+(U)\right| = 1$. Опираясь на (1.3), вычислим значения потоков по дугам $x_{i,j}$, которые допускают представление через известные коэффициенты $p_{i,j}$ и p_{i,v_i}, используя наблюдаемые величины $x_{i,v_i} = f_{i,v_i}$ и подставим их в исходную систему уравнений (1.1). Далее, из графа $G = (I,U)$ исключим те дуги, значения потоков по которым установлены, а также вершины, для которых известны величины внешних потоков $x_i = f_i$, где $i \in M \cap I^*$. Согласно [10], удалим из системы (1.3) те выражения, в которых отсутствуют неизвестные величины и введем величину q, обозначающую число оставшихся уравнений типа (1.3), содержащих неизвестные значения дуговых потоков. В результате всех операций получим модифицированный граф $\bar{G} = (\bar{I}, \bar{U})$, где $\bar{I}^* = I^* \setminus (M \cap I^*)$. Таким образом, система (1.1), (1.3) приобретает следующий вид [14]:

$$\sum_{j \in I_i^+(\bar{U})} x_{ij} - \sum_{j \in I_i^-(\bar{U})} x_{ji} = \begin{cases} a_i + x_i, i \in \bar{I}^* \\ a_i, i \in \bar{I} \backslash \bar{I}^* \end{cases} \tag{1.4}$$

$$\sum_{(i,j) \in \bar{U}} \lambda_{ij}^p x_{ij} = 0, \quad p \in P = \{1, \ldots, q\}, \text{ если } P \neq \varnothing. \tag{1.5}$$

Следует отметить, что множество P может оказаться пустым. В выражениях (1.5), связывающих неизвестные переменные, коэффициенты принимают следующие значения: $\lambda_{ij}^p = 1$, $\lambda_{i,v_i}^p = -p_{ij} p_{i,v_i}^{-1}$, $j \in I_i^+(\bar{U}) \setminus \{v_i\}$ при условии, что $\left|I_i^+(\bar{U})\right| > 1$ и $i \in \bar{I}$. В случае, если из узла i выходит единственная дуга, то $\lambda_{ij}^p = 0$, $|I_i^+(\bar{U})| = 1$, $i \in \bar{I}$.

В системах (1.4)–(1.5) в качестве неизвестных фигурируют потоки по дугам $x_{i,j}$, где $(i,j) \in \bar{U}$, а также значения внешних потоков x_i для узлов из $\bar{I}^*$, $\bar{I}^* \subseteq \bar{I}$. При этом орграф $\bar{G} = (\bar{I}, \bar{U})$ не обязательно является связным или двунаправленным. Более того, не исключена ситуация, при которой отдельные компоненты связности орграфа $\bar{G}$ не содержат в себе вершины из подмножества $\bar{I}^*$.

1.2 Формулировка задачи поиска оптимального решения

На основании описанных сведений, можно сформулировать задачу поиска оптимальной расстановки считывающих устройств в узлах графа $G = (I,U)$,

позволяющей просматривать всю сеть: "найти подмножество M узлов минимального размера, $M \subseteq I$, такое, чтобы соответствующая система (1.4)-(1.5) имеет единственное решение"[7, с. 70].

Теорема 1. *"Пусть A - матрица инцидентности связного ориентированного графа G. Строки усеченной матрицы инцидентности $\tilde{A}$, образованной удалением любой строки матрищь A, линейно независимы"* [7, с. 70].

Доказательство. Из теоремы в [8] получаем, что ранг A матрицы инцидентности равен $|I|-1$. "Дуги, соответствующие базисному множеству столбцов матрицы инцидентности, образуют остовное дерево графа G. Линейная независимость усеченной матрицы инцидентности $\tilde{A}$ следует из приведения подматрицы матрицы $\tilde{A}$, которая соответствует базисным столбцам, к виду верхней треугольной матрицы, определитель которой отличен от нуля. Теорема доказана"[7, с.70].

Согласно [7] строим вспомогательную матрицу H размерности $|I| \times (|I| - 1)$. Причиной этому - задача сократить количество столбцов $|U|$ в матрице инцидентности A ориентированного графа $G = (I, U)$ до величины $|I| - 1$. При построении вспомогательной матрицы используются описанные ранее параметры $\beta_{i,j}$, основанные на коэффицетах распределения потока p_{ij}, $(i, j) \in U$. Напомним, что на основании формулы (1.3) совокупный исходящий поток из вершины $i \in I$ представляется как сумма величин по исходящим дугам:

$$F(i) = \sum_{j \in I_i^+(U)} x_{i,j}$$

Из [7] возьмем построение матрицы H:

$$H_{i,j} = \begin{cases} \beta_{j,i}, \text{ если } e_j - \text{ каноническая дуга, } i \neq j, (j,i) \in U, \\ -\sum_{k \in I_i^+(U)} \beta_{i,k}, \text{ если } e_j - \text{ каноническая дуга, } i = j, \\ 0, \text{ если } i \text{ и } j \text{ не соединены дугой.} \end{cases}$$

Ранг матрицы H меньше $|I|$, поскольку данная матрица имеет линейно зависимые строки и сумма всех ее строк равна нулю.

Теорема 2. *"Строки матрицы $\tilde{H}$, образованной в результате удаления из матрицы H строки с номером $i^* \in \{1, \ldots, |I|\}$, являются линейно независимыми"* [7, с. 71].

Доказательство. Предположим, что ориентированный граф G имеет конечное число вершин и $|I| = n$. Пользуясь этим, пронумеруем все вершины: $1, \ldots, n$.

Выберем некоторую строку матрицы H под индексом i^* и положим $i^* = n$. Обозначим $\tilde{H}_1, \ldots, \tilde{H}_{n-1}$ строки матрицы $\tilde{H}$. Доказательство проведем от противного. Предположим, что в новообразовавшейся матрице $\tilde{H}$ строки $\tilde{H}_1, \ldots, \tilde{H}_{n-1}$ являются линейно зависимыми. Следовательно существуют некоторые коэффи-

циенты $b_1, \ldots, b_{n-1}, \sum_{i=1}^{n-1} b_i^2 > 0$, такие что для них выполняется равенство:

$$\sum_{i=1}^{n-1} b_i \tilde{H}_i = 0$$

Возьмем вектор $f^* = (f_{e_1}, \ldots, f_{e_n})$, составляющими которого являются значения дуговых потоков канонических дуги $(e_1, \ldots, e_n)$. Умножим последнее уравнения с обеих частей на данный вектор. Имеем:

$$b_1 \tilde{H}_1 f^* + \ldots + b_{n-1} \tilde{H}_{n-1} f^* = 0$$

Распишем данной уравнение подробнее, в результате чего получим:

$$b_1 \tilde{H}_{1,e_1} f_{e_1} + \ldots + b_1 \tilde{H}_{1,e_n} f_{e_n} + b_2 \tilde{H}_{2,e_1} f_{e_1} + \ldots + b_2 \tilde{H}_{2,e_n} f_{e_n} + \ldots$$
$$\ldots + b_{n-1} \tilde{H}_{n-1,e_1} f_{e_1} + \ldots + b_{n-1} \tilde{H}_{n-1,e_n} f_{e_n} = 0$$

По определению матрицы $\tilde{H}$ имеем:

$$\tilde{H}_{i,e_j} f_{e_j} = \beta_{j,i} f_{e_j} = f_{j,i}, i \neq j,$$

$$\tilde{H}_{i,e_j} f_{e_j} = -\left(\sum_{k \in I_i^+(U)} \beta_{i,k} \right) f_{e_j} = - \sum_{k \in I_i^+(U)} f_{i,k}, i = j,$$

Тогда уравнение для коэффицентов $b_1, \ldots, b_{n-1}$ принимает вид:

$$(\pm b_1 \mp b_2) f_{1,2} + \ldots + (\pm b_1 \mp b_{n-1}) f_{1,n-1} + \ldots$$
$$\ldots + (\pm b_{n-1} \mp b_1) f_{n-1,1} + \ldots + (\pm b_{n-1} \mp b_{n-2}) f_{n-1,n-2} +$$
$$+ b_1 (\pm f_{1,n} \mp f_{n,1}) + \ldots + b_{n-1} (\pm f_{n-1,n} \mp f_{n,n-1}) = 0$$

В данном уравнении каждое слагаемое состоит из значения потока $f_{i,j}$ и соответствующего коэффицента. $f_{i,j} = 0$ для случаев, когда i, j совпадают. Уравнение должно выполнятся для любых ненулевых значений $f_{i,j}$. Это означает, что либо $(\pm b_i \mp b_j) = 0, i < n, j < n$, либо $(\pm f_{k,n} \mp f_{n,k}) = 0, k = \overline{1, n-1}$. Положим $f_{k,n} = 0, f_{n,k} = 0, k = \overline{1, n-1}$. Для коэффицентов вида $(\pm b_i \mp b_j), i < n, j < n$ элементы имею противоположные знаки, так как зависят от направления дуги. Это приводит к тому, что данные коэффиценты также будут равны нулю. В результате имеем:

$$\begin{gathered} f_{i,j} = 0, i = j, \\ \pm b_i \mp b_j = 0, f_{i,j} \neq 0, i \neq j, i < n, j < n, \end{gathered} \tag{1.6}$$

$$b_k = 0, f_{k,n} \neq 0, \text{ или } f_{n,k} \neq 0, k < n. \tag{1.7}$$

В усечённой матрице инцидентности $\tilde{A}$ "каждый ее столбец содержит не более двух ненулевых элементов, которые равны $+1$ и -1. Пусть задана произвольная линейная комбинация строк матрицы $\tilde{A}$ с коэффициентами $b_1, \ldots, b_{n-1}$.

Тогда результатом этой комбинации будет вектор, чьи компоненты соответствуют левой части уравнений (1.6)–(1.7). Однако, согласно этим же уравнениям, каждая из таких компонент должна равняться нулю"[7, с.71]. Следовательно, выполняется равенство:

$$b_1\tilde{A}_1 + b_2\tilde{A}_2 + \cdots + b_{n-1}\tilde{A}_{n-1} = 0.$$

Из последнего уравнения вытекает, что строки матрицы $\tilde{A}$ линейно зависимы. Но это приводит к противоречию с утверждением теоремы 1, в которой говорится о линейной независимости этих строк. Получаем противоречие, что доказывает линейную независимость строк исходной матрицы $\tilde{H}$. Теорема доказана.

Исходя из двух описанных ранее теорем, можно сформулировать следующее следствие.

Следствие 1. *"Матрица, полученная путем удаления из $\tilde{H}$ столбца, соответствующего канонической дуге узла i^*, имеет размерность $(n-1)\times(n-1)$ и является матрицей полного ранга"* [7, с. 71].

Доказательство. Основано на теореме 2 и проводится аналогичным образом.

Теорема 3.*"Пусть $G = (I, U)-$ связный двунаправленный ориентированный граф, $p_{ij}, (i,j) \in U$ - известные коэффициенты разбиения потока и для множества I^* всех узлов графа G с внешним потоком выполняется: $I^* = \varnothing$. В качестве математической модели для идентификации потоковых характеристик на ненаблюдаемой части графа G используется недоопределенная система линейных алгебраических уравнений вида (1.1). Тогда достаточно одной просматриваемой дуги графа G для вычисления значений дуговых потоков всего графа"* [7, с. 72].

Доказательство. Выберем некоторый узел k и предположим, что соответствующая ему дуга e_k является обозреваемой и канонической. Для графа G сформируем усеченную матрицу $\tilde{H}$. Поскольку дуга e_k обозреваема, то для нее известно численное значение дугового потока f_{e_k}. Обозначим $\tilde{H}^{(k)}$ столбец, содержащий каноническую дугу e_k. Вычислим значения каждого элемента $\tilde{H}^{(k)}$, используя дополнительными уравнениями вида (1.3). Получим:

$$f_{k,j} = \beta_{k,j} f_{e_k} \tag{1.8}$$

Поскольку все значения столбца, содержащего элементы (1.8), известны, удалим его из матриц $\tilde{H}$. В результате получим матрицу $\widehat{H}$ размерности $(n-1)\times(n-1)$. Сформируем вектор правой части: $b = -\tilde{H}^{(k)}$. Составим следующую систему:

$$\widehat{H}x = b \tag{1.9}$$

"Решением данной системы является столбец x, элементами которого выступают значения потоков по дугам, выраженные через канонические дуги

(кроме дуги e_k). Поскольку матрица $\widehat{H}$ получена из $\tilde{H}$ путем удаления столбца $\tilde{H}^{(k)}$, воспользовавшись следствие 1, приведенным ранее, получаем, что матрица $\widehat{H}$ является матрицей полного ранга"[7, с. 72]. Это гарантирует, что система (1.9) имеет единственное решение, позволяющее восстановить значения потоков во всех дугах графа G с помощью (1.8). Теорема доказана.

Следствие 2. *"Пусть G - связный ориентированный двунаправленный граф, содержащий $k = |I^*|$ узлов с переменной интенсивностью. Математической моделью для данного графа является система вида (1.1), а $p_{ij}, (i, j) \in U$ - известные коэффициенты разбиения потока. Для определения численных значений дуговых потоков всего графа достаточно разместить $k = |I^*|$ сенсоров в узлах множества I^*"* [7, с. 72].

Доказательство. Пусть $v_1, \ldots, v_k$ - узлы с переменной интенсивностью. С помощью сенсоров получаем численные значения внешних потоков $f_i, i \in I^*$. Также, с помощью сенсоров, для k канонических дуг получаем численные значения дуговых потоков. Исходная система (1.1) содержит $n = |U|$ неизвестных, но после преобразований матрица $\tilde{H}'$ данной системы их станет $n - k$. Удалив столбцы $\tilde{H}'^{(\mathrm{v}_1)}, \ldots, \tilde{H}'^{(v_k)}$ из матрицы $\tilde{H}'$, сформируем вектор правой части: $b' = -\sum_{i=1}^{k} \tilde{H}'^{(v_i)}$. Получим матрицу $\widehat{H}'$ размерности $(n - k) \times (n - k)$. Также имеем систему:

$$\widehat{H'}x = b \tag{1.10}$$

"где x - вектор, содержащий значения дуговых потоков через канонические дуги, за исключением дуги e_k. По теореме 2 матрица $\tilde{H}'$ является матрицей полногой ранга, а значит, и $\widehat{H'}$ - матрица полного ранга. Таким образом, из системы (1.10) однозначно определяются значения дуговых потоков для канонических дуг"[7, с.72].

ГЛАВА 2. СВОЙСТВА СУБОПТИМАЛЬНЫХ РЕШЕНИЙ ЗАДАЧИ РАСПОЛОЖЕНИЯ СЕНСОРОВ ДЛЯ ОЦЕНКИ ПОТОКОВ НА НЕНАБЛЮДАЕМОЙ ЧАСТИ СЕТИ

2.1 Альтернативныый способ решения задачи

При решении задачи мониторинга сетевых потоков путем использования минимального числа сенсоров и установки их в определенные узлы из множества M, алгоритм полного перебора узлов графа G является неэффективным для сетей большой размерности. Причина этому - вычислительная сложность и временные затраты. Даже для относительно малой сети $G = (I, U), m = |I|$, затраты на перебор всех вариантов являются существенными [12].

$$C_m^1 = m, C_m^2 = \frac{m(m-1)}{2}, \ldots, C_m^k = \frac{m!}{k!(m-k)!}$$

При моделировании транспортных потоков в мегаполисах с числом автомобилей, превышающим сотни тысяч, возможны значительные изменения интенсивности движения на определенных участках (величинах дуговых потоков) и объемов въезжающего/выезжающего трафика (внешних потоков). В подобных условиях решения многих задач дискретной оптимизации демонстрируют устойчивость к незначительным изменениям входных параметров. Это позволяет вводить дополнительные ограничения, повышающие адекватность модели без существенных усложнений, и (или) наоборот, упрощать математические модели, пренебрегая малозначимыми ограничениями [12].

Поскольку рассматриваемая задача подразумевает использование информационных систем и измерений, для которых характерны ошибки округления, то, очевидно, что при решении будут присутствовать неточные данные. К ним относятся дуговые потоки $x_{ij}, (i, j)$, коэффициенты разбиения дуговых потоков $p_{ij}, (i, j) \in U$ и внешние потоки узлов $x_i, i \in I^*$. "Интервальные матрицы и векторы являются одним из эффективных методов представления неточных данных, обладающих с точки зрения пользователя рядом преимуществ"[7, с.73]. "Под интервальным вектором $\boldsymbol{b}$ будем понимать матрицу-столбец $\boldsymbol{b} = \{b : \underline{b} \leq b \leq \bar{b}\}$, т. е. $\boldsymbol{b} = [\underline{b}, \bar{b}] = [b_c - \delta, b_c + \delta]$, $\underline{b}, \bar{b} \in \mathbb{R}^m$, где $b_c = \frac{1}{2}(\underline{b} + \bar{b})$ - средний вектор (вектор середин); $\delta = \frac{1}{2}(\bar{b} - \underline{b})$ - неотрицательный вектор радиусов"[11, с.44].

Согласно [7] и следствию 2, приведенному ранее, установка $k = |I^*|$ сенсоров во всех вершинах множества $M = I^*$ графа G, гарантирует полную наблюдаемость сети. Значение $|M|$ обозреваемых узлов, при постановке в которые просматривается вся сеть, изменяется в отрезке $[\underline{h}, \bar{h}]$. При этом, $\underline{h} = 1$, а $\bar{h} = |I^*|, |M| \leq |I^*|$. Таким образом, на основании изложенного ранее способа, получаем, что верхняя граница $\bar{h}$ отрезка $[\underline{h}, \bar{h}]$ изменилась и стала равна $|I^*|$.

Данный результат существенно отличается от предыдущего показателя $\bar{h} = |I|$.

Для внешних потоков $x_i, i \in I^*$, введем новый параметр t, $t > 0$, называемый порогом интенсивности (intensity threshold). Допустимый диапазон его значений задается отрезком $[\underline{t}, \bar{t}]$, где $\underline{t} > 0$, а верхняя граница $\bar{t} = V$ определяется экспериментально. В практических задачах незначительные колебания внешнего потока x_i, $i \in I^*$, не приводят к значительным изменениям в решениях, что допускает введение порога интенсивности на отрезке $[\underline{t}, \bar{t}]$.

Зададим новое множество узлов $\tilde{I}^*, \tilde{I}^* \subseteq I^*$, выбрав случайное численное значение введенного ранее параметра $t \in [\underline{t}, \bar{t}]$. Для данного множества будут справедливы следующие неравенства:

$$|x_i| < t, i \in \tilde{I}^* \tag{2.1}$$

Изменим множество I^* на основании неравенств (2.1). Проведем это следующим образом: удалим из множества I^* узлов графа G узлы, для которых выполнены условия (2.1) с помощью соответствующей операции: $I^* = I^* \backslash \tilde{I}^*$. Исходя из вышеизложенного, получим, что для каждого узла из измененного множества I^* справедливо неравенство: $|x_i| \geq t, i \in I^*$.

После изменения исходных множеств I^* и $I \backslash I^*$ графа G, получим обновленную систему уравнений вида (1.1). На основании всех данных, собранных из сенсоров, и описанных ранее систем (1.2), (1.3), получаем разреженную систему линейных алгебраических уравнений вида (1.4) - (1.5). Решение данной системы имеет вид $x = \left(x_{ij}, (i,j) \in \bar{U}, x_i, i \in \bar{I}^*\right)$. После получения данного решения, сеть G будет полностью просматриваема, поскольку решение единственно.

2.2 Формулировка задачи поиска субоптимального решения

Исходя из описанного ранее варианта решения, можно "сформулировать новую задачу: задачу поиска субоптимальной (t-оптимальной) конфигурации установки сенсоров в узлах сети $G = (I, U)$. В [7] она звучит следующим образом: для заданного порога интенсивности $t, |x_i| \geq t, i \in \bar{I}^*$, найти такое подмножество M узлов графа G, $M \subseteq I$, чтобы соответствующая система (1.4)-(1.5) имела единственное решение"[7, с. 73]. В [7] также приведены численные результаты построения t-оптимального решения на различных примерах.

Изучение подобных решений, не являющихся оптимальными, имеет многолетнюю историю. Актуальность их изучения возрастает в условиях стремительного роста сложности и масштабов сетевых инфраструктур, развития вычислительных технологий - увеличения производительности процессоров, расширения объемов оперативной памяти, распространения многопроцессорных архитектур, а также благодаря появлению новых алгоритмов организации вычислений, поддерживаемых современными высокоуровневыми языками программирования и программными средствами [10]. Программные средства

обеспечивают комфортное и упрощенное взаимодействие с языками программирования, языки, в свою очередь, предоставляют огромный выбор различных библиотек и структур данных, облегчающих практическую реализацию алгоритмов.

Стоит отметить, что вычислительная сложность задачи зависит напрямую от количества рассматриваемых вариантов. Для данной задачи "естественным является следующий подход: перебирать t-оптимальные решения для различных численных значений порога интенсивности t из интервала $[\underline{t}, \bar{t}]$ до тех пор, пока не встретиться t-оптимальное решение, которое является приемлимым по числу сенсоров $|M| = |I^*|$ и значению t порога интенсивности. Далее для выбранного t-оптимального решения применяются стратегии случайного поиска с целью уменьшить числов $|M|$ обозреваемых узлов"[7, с.74]. В [7] приведены примеры построения t-оптимального численного решения для сетей, имеющих различное число узлов и дуг.

Таким образом, на основании приведенных ранее теоретических сведений, можно сделать вывод, что для задачи расстановки сенсоров с целью оценки потока сети поиск субоптимального решения (t-оптимального) является более предпочтительным за счет меньших вычислительных затрат и более простой реализации, нежели поиск оптимального решения. В последующих главах будут приведены конкретные примеры прострoения подобных решений, в том числе для одной и той же сети.

ГЛАВА 3. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ГЕНЕРАЦИИ СИЛЬНОСВЯЗНОГО НАПРАВЛЕННОГО ГРАФА

3.1 Выбор среды реализации

Для реализации программной части было выбрана система компьютерной алгебры Wolfram Mathematica. Это решение обусловлено ее функциональными возможностями, подходящими для задач, связанных с аналитическими вычислениями, визуализацией данных и построением графов.

Одним из ключевых преимуществ Wolfram Mathematica являются мощные средства символьных вычислений. Программа предоставляет обширный инструментарий для символьной обработки выражений, решения уравнений и систем, а также их упрощения.

Кроме того, Wolfram Mathematica обладает встроенной поддержкой работы с графами – как ориентированными, так и неориентированными, что позволяет эффективно реализовать построение и анализ графовых структур, визуализировать траектории, связи и другие элементы, используемые в алгоритмах. Наглядность и гибкость графической интерпретации данных значительно облегчают понимание промежуточных и конечных результатов.

Отдельное внимание заслуживает язык Wolfram Language, разработанный специально для работы с математикой, данными и алгоритмами. Он сочетает декларативный стиль с широкими возможностями абстракции, что позволяет быстро реализовывать сложные идеи, работать с символами, списками, графами и другими структурами. Язык предоставляет множество встроенных функций, упрощающих реализацию вычислений, логики, анализа и визуализации без необходимости в низкоуровневом программировании. Все это позволяет компактно описывать математические модели и процессы.

3.2 Назначение и цели разработки

В рамках данной работы была разработана программа, предназначенная для генерации случайного направленного графа, удовлетворяющего определённым структурным ограничениям. Полученный граф сохраняется в текстовый файл в заранее заданном формате. В дальнейшем, в ходе программной реализации поиска субоптимального решения, полученный граф служит входными данными, которыми манипулирует программа. Эти данные используются для решения недоопределенной разряженной линейной системы уравнений и ее поэтапной визуализации при помомщи встроенных средств системы компьютерной алгебры Wolfram Mathematica.

Целью разработки является генерация случайного ориентированного графа с заданным количеством вершин $|I|$ и рёбер $|U|$. Граф называется ориентированным, если его ребрам присвоены направления [6]. Именно такой граф

используется в дальнейшем при решении задачи поиска расстановки сенсоров, так как он наиболее точно отражает природу потоков и взаимодействий в сетевых системах. В [5] сказано, что, зачастую, в потоковых задачах фигурируют уравнения баланса потоков в узлах, что дополнительно подчеркивает необходимость генерации именно ориентированного графа. Также необходимо, чтобы генерируемый граф содержал 1 компоненту связности. В противном случае, это может привести к наличию висячих вершин, отсутствию решения за счет наличия независимых подсистем и другим проблемам. Таким образом, генерируемый граф должен соответствовать следующим условиям:

- в графе не должно быть петель, то есть рёбер вида (v, v);
- запрещены повторяющиеся рёбра;
- не допускаются антипараллельные рёбра: если существует ребро (u, v), то ребро (v, u) быть не может;
- каждая вершина получает маркировку 0 либо x, причём количество меток x задаётся отдельно;
- генерируемый граф должен был быть сильносвязным.

Последнее условие не является обязательным в контексте задачи, поскольку далее, в последующих главах, мы будем работать с двунаправленной сетью. Однако на практике генерация сильносвязного ориентированного графа реализуется проще нежели генерация слабосвязного, а сама задача не требует строго наличия слабосвязнозного орграфа.

3.3 Описание алгоритма генерации графа

Процесс генерации направленного графа осуществляется в несколько этапов. Первым этапом является инициализация параметров. Пользователь задаёт следующие параметры:

- количество вершин $|I|$;
- количество рёбер $|U|$;
- количество вершин с маркировкой x (вершины с внешним потоком).

```
(*Параметры*)
vertexCount = 11;
edgeCount = 13;
numX = 6;
SeedRandom[];
```

Далее происходит формирование набора ребер. Первоначально, формируется структура из $|I|$ ребер графа. Основой данной структуры выступает ориентированный цикл, включающий в себя все вершины из множества I, т.е. гамильтонов цикл[6]. Такой подход гарантирует генерацию сильносвязного орграфа, поскольку каждая вершина имеет как минимум одну входящую и одну исходящую дуги. Остальные $|U|$ - $|I|$ ребер добавляются отдельно. Выбираются 2 случайные вершины из множества вершин и соединяются ребром, таким образом формирую проверяемое далее ребро. Каждый кандидат на добавление проверяется на соответствие следующим условиям:

- вершины начала и конца не совпадают (отсутствие петель);
- ребро не должно уже существовать в графе;
- не допускается существование обратного (антипараллельного) ребра.

Каждое потенциальное ребро, прошедшее все проверки, добавляется во множество U. Подобный подход генерации позволяет не только задать его основные параметры, но и гарантирует выполнение логических и структурных ограничений, обеспечивающих пригодность графа для последующего использования в задаче. После добавления всех ребер, полученное множество проходит дополнительные проверки, описанные далее. Код генерации ребер графа приведен ниже.

```
(*1.Строим сильно связный граф как направленный цикл*)
baseEdges = Table[{i, Mod[i, vertexCount] + 1},
{i, 1, vertexCount}];
(*2.Добавим дополнительные рёбра, соблюдая условия*)
additionalEdgesNeeded = edgeCount − vertexCount;
existingSet =< || >; (*быстрое хранилище направленных рёбер*)
Do[existingSet[{e[[1]], e[[2]]}] = True, {e, baseEdges}];
extraEdges = {};
While[Length[extraEdges] < additionalEdgesNeeded,
u = RandomInteger[{1, vertexCount}];
v = RandomInteger[{1, vertexCount}];
If[u == v, Continue[]];
(*запрет петель*)
If[!KeyExistsQ[existingSet, {u, v}]&&!
KeyExistsQ[existingSet, {v, u}]&&!
MemberQ[extraEdges, {u, v}]&&!
MemberQ[extraEdges, {v, u}],
AppendTo[extraEdges, {u, v}];
```

existingSet[{u, v}] = True;];];

Затем случайным образом выбираются k различных вершин, которые получают маркировку x. Остальные вершины получают значение 0. Результат работы программы сохраняется в текстовый файл, который размещается в той же директории, что и исполняемый файл программы.

Этап формирование текстового файла является крайне важным. После генерации, полученные данные не обладают необходимой структурой и должны быть сгруппированны таким образом, чтобы основная программа смогла прочитать его и выполнить необходимые расчеты. Этап упорядочивание и преобразования данных выполняется перед сохранением в файл.

Структура файла выглядит следующим образом:

- первая строка — количество вершин;
- вторая строка — количество рёбер;
- список рёбер в формате {источник, сток};
- маркировка вершин (0 или x).

Код программы, отвечающий за случайный выбор вершин с маркировкой x и формирование выходного файла, выглядит следующим образом:

```
(*Выбор случайных вершин с x*)
xVertices = RandomSample[Range[vertexCount], numX];
(*Формируем строки*)
header = {"/*|I|*/ > ToString[vertexCount],
"/*|U|*/ > ToString[Length[allEdges]]};
edgesStr = Map["{ > ToString[#[[1]]] <>  >
ToString[#[[2]]] <> "}"&, allEdges];
bValues = Table["/*b_ > ToString[i] <> " */ >
If[MemberQ[xVertices, i], "x "0"],
{i, 1, vertexCount}]; outputLines = Join[header, edgesStr, bValues];
(*Сохраняем в ту же папку, что и .nb файл*)
notebookDir = NotebookDirectory[];
Export[FileNameJoin[{notebookDir, "graph.txt"}], outputLines, "Text"];
```

3.4 Проверка полученных результатов

Для обеспечения надёжности генерации программа в процессе своего выполнения использует встроенные проверки.

Первая из них - отсутствие петель: перед добавлением нового ребра (u, v) проверяется условие:

$$u \neq v$$

Это исключает возможность появления петель — рёбер, начинающихся и заканчивающихся в одной и той же вершине [6]. Такие рёбра не соответствуют условиям задачи и отбрасываются на этапе генерации.

Следующая проверка предотвращает добавления одинаковых рёбер. Для этого используется структура данных ассоциативный массив (или хеш-таблица), выступающая в роли множества уже добавленных рёбер. Отличительная особенность данной структуры в том, что все элементы в ней являются уникальными элементами. Происходит это благодаря использованию хеш-функции [6]. Данная коллекция идеально подходит в данной ситуации, так как при попытке добавить новое ребро (u, v) программа проверяет:

$$(u, v) \notin U$$

Ребро-кандидат помещают в хеш-функцию и ее результат сравнивают с содержимым ассоциативного массива. Если в массиве уже существует элемент со схожей хеш-функцией, то ребро-кандидат игнорируется [6]. Это обеспечивает уникальность каждого ребра и отсутствие дубликатов.

Далее дополнительно исключается наличие обратных (антипараллельных) рёбер. Для каждого кандидата на добавление (u, v) выполняется проверка:

$$(v, u) \notin U$$

Таким образом, если в графе уже есть ребро (u, v), то ребро (v, u) не может быть добавлено. Проверка также осуществляется с помощью ассоциативного массива, содержащего все текущие рёбра.

Последней проверкой является построение графа. На основе сгенерированных данных строится граф и проверяется, является ли он связным. Если граф не является связным, программа выводит соответствующее сообщение, уведомляя пользователя. Ниже приведена реализация данных проверок:

```
allEdges = Join[baseEdges, extraEdges];
If[Length[DeleteDuplicates[allEdges]] =! = Length[allEdges],
Print["Повторяющиеся рёбра!"]];
If[AnyTrue[allEdges, MemberQ[allEdges, Reverse[#]]&],
Print["Найдены антипараллельные рёбра!"]];
graph = Graph[allEdges, DirectedEdges → True];
Graph[graph, EdgeStyle -> Directive[Black, Thick],
VertexLabels -> Placed["Name Center],
VertexSize -> 0.5, VertexStyle -> Directive[EdgeForm[Thick], White],
```

```
VertexLabelStyle -> Directive[Black, 24],
GraphLayout -> "CircularEmbedding
VertexShapeFunction -> {xx_ :> If[MemberQ[xVertices, xx], "Square "Circle"]}]
If[!StronglyConnectedGraphQ[graph], Print["Граф не сильно связный!"]];
```

3.5 Визуализация полученных результатов

Для проведения анализа и оценки полученных данных в процессе работы программы происходит их визуализация. Визуализация выполняется при помощи ориентированного графа, содержащего все ребра, полученные в ходе генерации. Все вершины графа нумеруются. Это обеспечивает однозначную идентификацию каждой вершины и упрощает чтение изображения, особенно при большой размерности графа. Пример работы визуализации показан на рисунке 3.1.

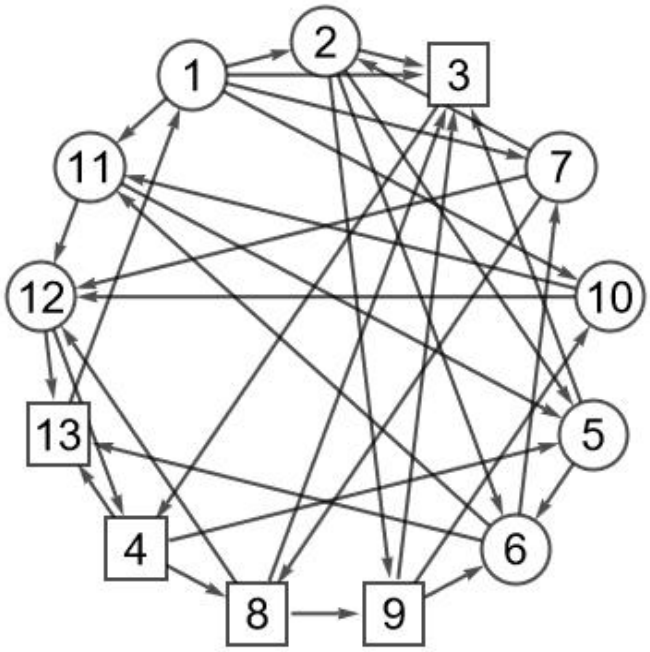

Рисунок 3.1 – Визуализация полученных данных

Таким образом, разработанная программа позволяет эффективно и гибко генерировать сильносвязные ориентированные графы с заданными ограничениями. Благодаря возможности управления параметрами и автоматическим проверкам, получаемые графы можно использовать в качестве входных данных для поиска субоптимального решения. Алгоритм их генерации гарантирует связность, отсутствие повторяющихся ребер, антипараллельных ребер и петель, а последующая визуализация позволяет проверить полученные результаты и оценить их. Код программы в полном объеме расположен в Приложении В.

ГЛАВА 4. ПРОГРАММАНАЯ РЕАЛИЗАЦИЯ ПОСТРОЕНИЯ СУБОПТИМАЛЬНОГО РЕШЕНИЯ

4.1 Получение входных данных

Программа поиска субоптимального решения, как и программа генерации сильносвязного графа, реализована в среде компьютерной алгебры Wolfram Mathematica. Входные данные хранятся в текстовом файле, формат которого был описан ранее.

Программа считывает данные построчно и на их основе формирует граф $G = (I, U)$, где I - множество вершин, U - множество ребер. Так как исходный текстовый файл не содержит обратных ребер, то при формировании графа, программа добавляет для каждого ребра обратное, что позволяет сформировать двунаправленную сеть. Такая трансформация обусловлена актуальность подобных сетей в прикладных задачах. Процедура чтения и первоначальной обработки данных реализована в виде отдельной функции. Данный подход будет использоваться и далее, поскольку он обеспечивает модульность и удобство повторного использования в других частях программы. Реализованная функция не только считывает данные из файла, но и проводит первичную валидацию данных. Код, используемый для данного шага работы программы, представлен ниже:

```
readGraph2(f_, d_) := Module[
{fn = FileNameJoin[{d, f}], s, im, um, u, b},
s = OpenRead[fn];
im = Read[s, {Word, Number}][[2]];
um = Read[s, {Word, Number}][[2]];
u = Flatten[({#1[[1]] ↔ #1[[2]], #1[[2]] ↔ #1[[1]]}&)/@
ReadList[s, Expression, um]];
b = ConstantArray[0, im];
Apply[(b[[Read[StringToStream[StringTake[#1, {5, -3}]],
Number]]] = #2)&,
ReadList[s, {Word, Expression}, im], {1}];
{Graph[u, VertexSize → Medium, VertexLabels → Placed[Name, Center],
VertexStyle → Directive[White],
VertexShapeFunction → {xx_ :> If[b[[xx]] === x, Square, Circle]},
VertexLabelStyle → Directive[Black, 24], GraphLayout → CircularEmbedding], b}]
(vert = VertexList[g]; )
(newVert = Sort[vert]; )
(gr = Graph[newVert, EdgeList[g]]; )
({g, b} = readGraph2("graph.txt, NotebookDirectory[]); )
(b = (MapIndexed[#1/.x → x_{#2[[1]]}&, b]
```

[[#1]]&)/@VertexList[gr];)
GraphPlot[g, EdgeStyle → Directive[Black, Thick],
VertexLabels → Placed[Name, Center], VertexSize → Large,
VertexStyle → Directive[EdgeForm[Thick], White], MultiedgeStyle → 0.07]

4.2 Формирование системы уравнений

После первоначальной обработки входных данных и их визуализации, программа переходит к следующему шагу - формированию системы линейных уравнений, отражающей закон сохранения потока в каждой вершине графа. Полученная система имеет вид (1.1).

Формирование происходит на основе структуры ориентированного графа, проводится обход каждой вершины графа и просматриваются все инцидентные ребра - как входящие, так и исходящие. Исходящие из вершины ребра добавляются в систему со знаком плюс, входящие в вершину ребра добавляются со знаком минус. В результате получаем систему из $|I|$ уравнений. Для вершин с некоторым внешним потоком, правая часть соответствующего уравнения не равна нулю, так как их внешний поток известен за счет установки сенсоров.

Функция, формирующая систему, и визуализация входных данных представлены ниже.
(vert = VertexList[g];)
(newVert = Sort[vert];)
(gr = Graph[newVert, EdgeList[g]];)
(b = (MapIndexed [#1/. $x \to x_{\#2[[1]]}$&, b] [[#1]]&) /@VertexList[gr];)
GraphPlot[g, EdgeStyle → Directive[Black, Thick],
VertexLabels → Placed[Name, Center], VertexSize → Large,
VertexStyle → Directive[EdgeForm[Thick], White], MultiedgeStyle → 0.07]
balanceEqs(balanceEqs = (Total[($x_{\#1}$&) /@
EdgeList[g, #1_]] − Total[($x_{\#1}$&)
/@EdgeList[g, _#1]] = b[[#1]]&)/@VertexList[gr];)

4.3 Преобразования системы уравнений

После формирования исходной системы уравнений, в соответствии с начальными данными, устанавливаются сенсоры во все узлы множества M и происходит сбор информации. При этом соответствующие $x_{i,j}$ заменяются на $f_{i,j}$, в соответствии с (1.2).

На основании полученной системы уравнений в исходном графе происходит поиск ребер с известными дуговыми потоками. После получения всех подобных ребер, они удаляются из графа и полученный граф визуализируется.
M = Select[Range[Length[b]], b[[#]] =! = 0&];

```
Print["M = M];
b = b/.x → f;
Print["b = b];
balanceEqs = ((Total[If[Or[MemberQ[M, #[[1]]], MemberQ[M, #[[2]]]],
Subscript[f, #], Subscript[x, #]]&/@
EdgeList[g, #DirectedEdge_]] − Total[If[Or[MemberQ[M, #[[1]]],
MemberQ[M, #[[2]]]], Subscript[f, #],
Subscript[x, #]]&/@EdgeList[g, _DirectedEdge#]]))
== b[[#]]&/@VertexList[gr];
TableForm[balanceEqs]
incL = (IncidenceList[gr, #]&/@M)//Flatten;
incL1 = DeleteDuplicates[incL];
Print["Ребра для удаления: incL1];
newGraph = VertexDelete[g, M];
GraphPlot[newGraph,
EdgeStyle → Directive[Black, Thick],
VertexStyle → Directive[EdgeForm[Thick], White], MultiedgeStyle → 0.05,
VertexLabels → Placed["Name Center], VertexSize → 0.25,
VertexLabelStyle → Directive[Black, 20], MultiedgeStyle → 0.07]
```

Следующим шагом является поиск дуг, образующих разрез исходного графа относительно множества M. Полученное множество также визуализируется путем использования функции Print[].

Из множества дуг разреза формируется подмножество дуг путем удаления дубликатов по следующему правилу: два ребра считаются одинаковыми, если у них совпадают первые элементы (исходные вершины). При помощи данного множества дуг происходит формирование дополнительных уравнений вида (1.3).

```
M' = Complement[VertexList[g], M];
edg = EdgeList[g];
M+ = Select[edg, MemberQ[M', #[[1]]]&&
MemberQ[M, #[[2]]]&];
Print["M = M];
For[i = 1, i ≤ Length[M+], i++,
Print[M+, i, "] = M+[[i]],
t | tM+[i,"][1] = M+[[i]][[1]],
t | tM+[i,"][2] = M+[[i]][[2]]];
Print["Length[M+] = Length[M+];
Print["Length[M] = Length[M]];
edgesForAdditionalEquations = DeleteDuplicates[M+],
#1[[1]] == #2[[1]]&];
```

Дополнительные уравнения подставляются в исходную систему, в резуль-

тате чего неизвестных становится меньше. В соответствии с новой системой также удаляются ребра графа с известными дуговыми потоками, а полученный граф визуализируется. Данный граф содержит в себе неизвестные, которые будут найдены путем решения соответствующей ему системы уравнений.

```
g1 = Fold[EdgeDelete[1, uDirectedEdgev/; u == 2]&,
newGraph, #[[1]]&/@[M+]];
GraphPlot[g1, EdgeStyle → Directive[Black, Thick],
VertexStyle → Directive[EdgeForm[Thick], White],
MultiedgeStyle → 0.05,
VertexLabels → Placed["Name Center], VertexSize → 0.25]
```

4.4 Решение системы

При решении системы используются 4 основные функции:

- **getStartReplacement[Subscript[x, i_ -> j_]]** - создаёт замену для конкретного ребра (i, j);
- **replaceX[expr_]** - заменяется все $x_{i,j}$ на выражения вида (1.3) при первоначальном сборе информации;
- **getSubstitution[edge_]** - среди всех решений, находит решение для определенного ребра графа;
- **replaceDynamicX[expr_]** - заменяется $x_{i,j}$ на выражения вида (1.3), используя уже найденные решения.

Реализация данных функций выглядит следующим образом:

```
replaceX[expr_] := expr/.{Subscript[x, i_DirectedEdgej_] :>
With[{k = Select[edgesForAdditionalEquations, #[[1]] == i&]},
If[k =! = {},
Module[{repl},
repl = (Subscript[f, iDirectedEdgek[[1]][[2]]] ∗ Subscript[p,
iDirectedEdgej])/Subscript[p, iDirectedEdgek[[1]][[2]]];
repl],
Subscript[x, iDirectedEdgej]]}];
getStartReplacement[Subscript[x, i_DirectedEdgej_]] := With[
{k = Select[edgesForAdditionalEquations, #[[1]] == i&]},
If[k =! = {},
Module[{repl},
repl = (Subscript[f, iDirectedEdgek[[1]][[2]]] ∗ Subscript[p,
iDirectedEdgej])/Subscript[p, iDirectedEdgek[[1]][[2]]];
repl],
Subscript[x, iDirectedEdgej](∗если подходящего k нет*)]};
```

```
getSubstitution[edge_] :=
SelectFirst[solutions, MatchQ[First[#], Subscript[x, edge]]&];
replaceDynamicX[expr_] :=
expr/.{Subscript[x, i_DirectedEdge j_] :>
With[{k = Select[newAdditionalEdges, #[[1]] == i&]},
If[k =! = {},
Module[{repl},
sub = getSubstitution[iDirectedEdgek[[1]][[2]]];
repl = (sub[[2]] * Subscript[p, iDirectedEdgej])/Subscript[
p, iDirectedEdgek[[1]][[2]]];
AppendTo[solutions, Subscript[x, iDirectedEdgej] → repl];
DeleteDuplicates[solutions]; repl], Subscript[x, iDirectedEdgej]}]};
```

Решение системы уравнений происходит в виде цикла. Цикл работает до тех пор, пока в системе есть неизвестные $x_{i,j}$. Из всей системы выбираются уравнения, содержащие неизвестные, и решаются. Полученные дуговые потоки добавляются к списку решений, а ребра, соответствующие данным дуговым потокам, добавляются во множество, на основе которых формируются уравнения (1.3). Последним шагом итерации выполняется подстановка решений в исходную систему уравнений. После этого цикл повторяет свою работу по описанному ранее алгоритму.

```
balanceEqs//forma;
balanceEqs = replaceX[balanceEqs];
systemWithData = balanceEqs;
uniqueX = DeleteDuplicates@
Select[Level[balanceEqs, {0, Infinity}],
MatchQ[#, Subscript[x, _DirectedEdge_]]&];
replacements = Table[el → getStartReplacement[el], {el, uniqueX}];
replacements =
Select[replacements, !
MatchQ[#[[2]], Subscript[x, _DirectedEdge_]]&];
Print["Подстановка дополнительных уравнений"]
systemWithData//forma
(*основной цикл программы*)
While[AnyTrue[systemWithData, !FreeQ[#, x]&],
(*Ищем решение на уравнениях, где слева есть x*)
eqWithX =
Select[systemWithData, !
FreeQ[First[#], x]&]; (*уравнения с X в левой части*)
vars = DeleteDuplicates@
Cases[eqWithX, Subscript[x, _DirectedEdge], Infinity];
sol = Reduce[eqWithX, vars];
```

```
If[Head[sol] === Or,
sol = First[sol]];
xSolutions =
Select[sol, !FreeQ[First[#], x]&&FreeQ[Last[#], x]&];
xSolutions = List@@ToRules[xSolutions];
Print[xSolutions];
(*Добавляем новые ребра и решения*)
solutions = Join[solutions, xSolutions];
newEdges = (First[#]/.
Subscript[x, i_DirectedEdge j_] :> iDirectedEdge j)&/@
xSolutions;
newAdditionalEdges = Join[newAdditionalEdges, newEdges];
(*Подставляем решения в систему уравнений*)
systemWithData = systemWithData/.solutions;
systemWithData = replaceDynamicX[systemWithData]; ];
```

По завершению работы выполняется визуализация решений посредством команды Print[] и проверка полученных результатов. В систему, полученную после сбора информации, подставляются все сформированные до начала работы цикла уравнения вида (1.3) и решения системы, полученные в результате работы цикла. При помощи функции Simplify[] эта система сравнивается с системой, преобразованной при работе цикла.

```
Print["Итоговая система после всех подстановок"];
systemWithData//forma
Print["Решения"];
Print[TableForm[DeleteDuplicates[solutions]]];
Print["Simplify:
Simplify[systemWithData == (balanceEqs/.replacements)/.solutions]]];
```

ГЛАВА 5. ПРИМЕР ОПТИМАЛЬНОГО РЕШЕНИЯ ЗАДАЧИ ОЦЕНКИ ПОТОКА НА НЕНАБЛЮДАЕМОЙ ЧАСТИ ДВУНАПРАВЛЕННОЙ СЕТИ

5.1 Визуализация исходной двунаправленной сети

На рисунке 5.1 представлен конечный связный двунаправленный граф $G = (I, U)$, где

$$I = \{1, 2, 3, 4, 5, 6, 7, 8\},$$

$$I^* = \{2, 3, 6, 7\},$$

$$U = \{(1, 2), (1, 5), (2, 1), (2, 6), (3, 4),$$

$$(3, 6), (4, 3), (4, 7), (4, 8), (5, 1),$$

$$(6, 2), (6, 3), (6, 7), (7, 4), (7, 6), (8, 4)\}.$$

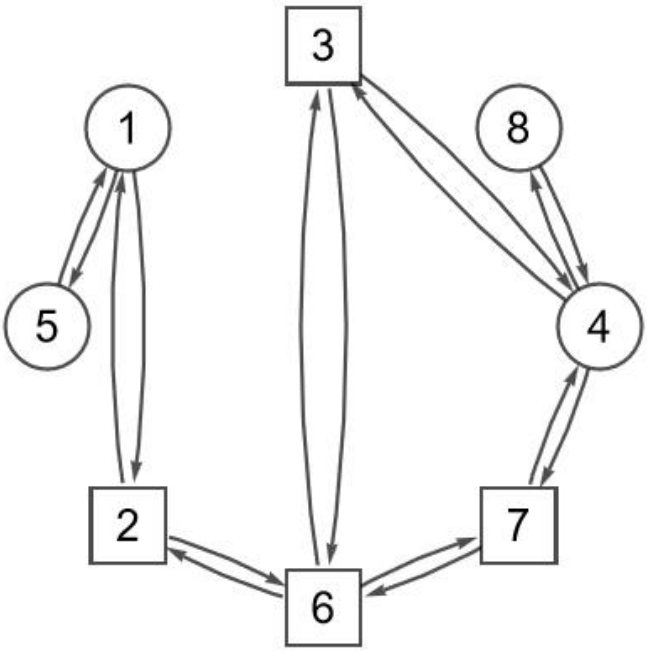

Рисунок 5.1 – Конечный связный двунаправленный граф G

Построим аналитическое и численное решения разряженной системы линейных уравнений типа (1.1) для случая, когда сенсор установлен в узле $M = \{6\}$ для графа G, изображенного на рисунке 5.1.

5.2 Аналитическое решение

Для графа G, изображенного на рисунке 5.1, рассмотрим систему линейных алгебраических уравнений:

$$\begin{gathered}
x_{1,2} + x_{1,5} - x_{2,1} - x_{5,1} = 0, \\
x_{2,1} + x_{2,6} - x_{1,2} - x_{6,2} = x_2, \\
x_{3,4} + x_{3,6} - x_{4,3} - x_{6,3} = x_3, \\
x_{4,3} + x_{4,7} + x_{4,8} - x_{3,4} - x_{7,4} - x_{8,4} = 0, \\
x_{5,1} - x_{1,5} = 0, \\
x_{6,2} + x_{6,3} + x_{6,7} - x_{2,6} - x_{3,6} - x_{7,6} = x_6, \\
x_{7,4} + x_{7,6} - x_{4,7} - x_{6,7} = x_7, \\
x_{8,4} - x_{4,8} = 0.
\end{gathered} \tag{5.1}$$

Предположим, что множество контролируемых вершин для графа G, изображенного на рисунке 5.1, равно $M = \{6\}$. Обозначим разрез графа G относительно заданного множества узлов M через $CC(M)$. Построим разрез $CC(M)$ для графа G относительно множества $M = \{6\}$.

Сформируем множества:

$$\begin{gathered}
M^+ = I(CC(M)) \setminus M = \{2, 3, 7\}, \\
M^* = M \cup M^+ = \{6\} \cup \{2, 3, 7\} = \{2, 3, 6, 7\}, \\
I \setminus M^* = \{1, 4, 5, 8\}.
\end{gathered}$$

Значения дуговых потоков по всем входящим и исходящим дугам для каждого узла i из множества $M = \{6\}$ (контролируемый узел) известны:

$$\begin{gathered}
x_{6,2} = f_{6,2}, \quad x_{2,6} = f_{2,6}, \\
x_{6,3} = f_{6,3}, \quad x_{3,6} = f_{3,6}, \\
x_{6,7} = f_{6,7}, \quad x_{7,6} = f_{7,6}.
\end{gathered}$$

Помимо этого, нам также известны значения $x_i = f_i,\ i \in M \bigcap I^*$:

$$x_6 = f_6.$$

Имеем:

$$\begin{gathered}
x_{6,2} = f_{6,2}, \quad x_{6,3} = f_{6,3}, \\
x_{6,7} = f_{6,7}, \quad x_{2,6} = f_{2,6}, \\
x_{3,6} = f_{3,6}, \quad x_{7,6} = f_{7,6}, \\
x_6 = f_6.
\end{gathered} \tag{5.2}$$

Подставим (5.2) в систему линейных уравнений (5.1). Удалим из графа $G = (I,\ U)$ множество дуг, на которых потоки по дугам известны в соответствии с (5.2). Также удалим из графа G множество вершин $i\ M$, $M = \{6\}$. Мы получаем граф G', показанный на рисунке 5.2.

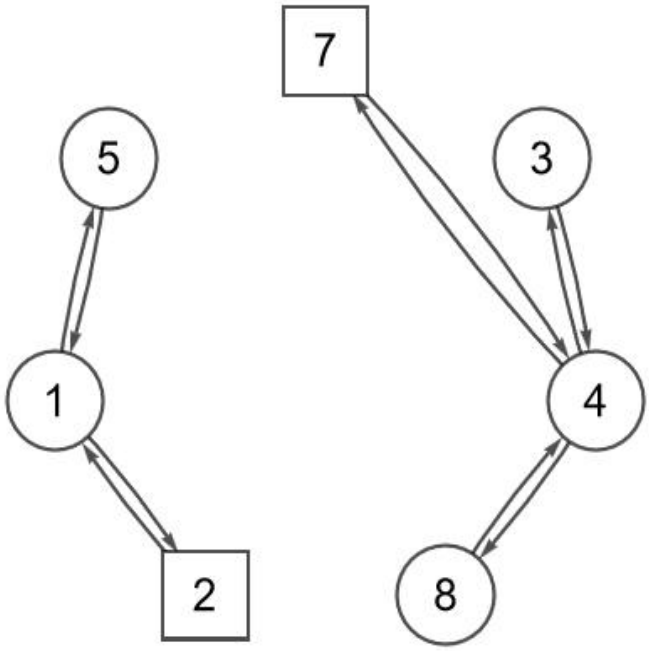

Рисунок 5.2 – Граф G'

Остальные дуговые потоки для ребер, исходящих из узлов множества $M^+ = I(CC(M)) \setminus M = \{2, 3, 7\}$, могут быть выражены из потока дуг, исходящих из множества M^+.

Получаем следующие уравнения:

$$\begin{gathered} x_{2,1} = \frac{p_{2,1}}{p_{2,6}} x_{2,6}, \\ x_{2,6} = f_{2,6}, \\ x_{3,4} = \frac{p_{3,4}}{p_{3,6}} x_{3,6}, \\ x_{3,6} = f_{3,6}, \\ x_{7,4} = \frac{p_{7,4}}{p_{7,6}} x_{7,6}, \\ x_{7,6} = f_{7,6}. \end{gathered} \tag{5.3}$$

На рисунке 5.3 представлен граф $\overline{G} = (\overline{I},\, \overline{U})$ полученный путем удаления множества ребер, на которых дуговой поток известен в соответствии с (5.3) для графа G' (см. рисунок 5.2). Система (5.1) для графа $\overline{G}$ (см. рисунок 5.3) преобразуется к виду (6.4).

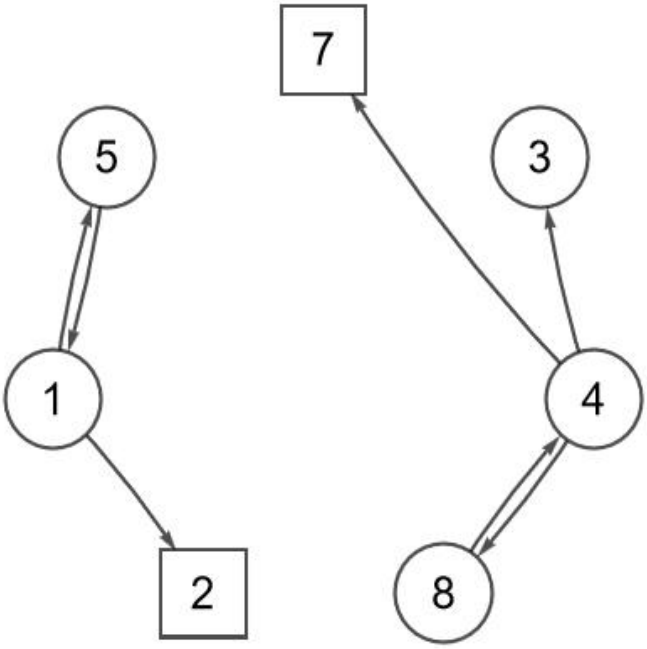

Рисунок 5.3 – Граф $\overline{G}$

$$x_{1,2} + x_{1,5} - \frac{p_{2,1}}{p_{2,6}} f_{2,6} - x_{5,1} = 0,$$

$$\frac{p_{2,1}}{p_{2,6}} f_{2,6} + f_{2,6} - x_{1,2} - f_{6,2} = x_2,$$

$$\frac{p_{3,4}}{p_{3,6}} f_{3,6} + f_{3,6} - x_{4,3} - f_{6,3} = x_3,$$

$$x_{4,3} + x_{4,7} + x_{4,8} - \frac{p_{3,4}}{p_{3,6}} f_{3,6} - \frac{p_{7,4}}{p_{7,6}} f_{7,6} - x_{8,4} = 0, \tag{5.4}$$

$$x_{5,1} - x_{1,5} = 0,$$

$$f_{6,2} + f_{6,3} + f_{6,7} - f_{2,6} - f_{3,6} - f_{7,6} = f_6,$$

$$\frac{p_{7,4}}{p_{7,6}} f_{7,6} + f_{7,6} - x_{4,7} - f_{6,7} = x_7,$$

$$x_{8,4} - x_{4,8} = 0.$$

Представим систему (6.4) в виде (5.5).

$$\begin{gathered} x_{1,2} + x_{1,5} - x_{5,1} = b_1, \\ -x_{1,2} = x_2 + b_2, \\ -x_{4,3} = x_3 + b_3, \\ x_{4,3} + x_{4,7} + x_{4,8} - x_{8,4} = b_4, \\ x_{5,1} - x_{1,5} = 0, \\ -x_{4,7} = x_7 + b_7, \\ x_{8,4} - x_{4,8} = 0. \end{gathered} \tag{5.5}$$

где

$$b_1 = \frac{p_{2,1}}{p_{2,6}} f_{2,6}, \;\; b_2 = -\frac{p_{2,1}}{p_{2,6}} f_{2,6} - f_{2,6} + f_{6,2},$$
$$b_3 = -\frac{p_{3,4}}{p_{3,6}} f_{3,6} - f_{3,6} + f_{6,3}, \;\; b_4 = \frac{p_{3,4}}{p_{3,6}} f_{36} + \frac{p_{7,4}}{p_{7,6}} f_{7,6},$$
$$b_5 = 0, \;\; b_7 = -\frac{p_{7,4}}{p_{7,6}} f_{7,6} - f_{7,6} + f_{6,7}, \;\; b_8 = 0.$$

Потоки дуг $x_{i,j}$, (i,j) $\overline{U}$, соответствующие дугам, исходящим из множества узлов $I \setminus M^* = \{1,4,5,8\}$, неизвестны. Для данных неизвестных потоков $x_{i,j}$ составим дополнительные уравнения вида (1.5), $(i,j) \in \overline{U}$.

Выберем произвольную исходящую дугу, которая начинается в узле i множества $I \setminus M^*$, где $I \setminus M^* = \{1,4,5,8\}$, если это возможно, например, для узла $i = 1$ выбираем дугу $(1,2)$. Тогда для всех исходящих дуг из узла $i = 1$, кроме $(1,2)$, верно равенство:

$$x_{1,5} = \frac{p_{1,5}}{p_{1,2}} x_{1,2} \tag{5.6}$$

Выберем произвольную исходящую дугу, которая начинается в узле $i = 4$ множества $I \setminus M^* = \{1,4,5,8\}$, например,$(4,8)$. Тогда для дуги $(4,3)$, исходящей из узла $i = 4$, и дуги $(4,7)$ справедливы равенства:

$$x_{4,3} = \frac{p_{4,3}}{p_{4,8}} x_{4,8}, \quad x_{4,7} = \frac{p_{4,7}}{p_{4,8}} x_{4,8}. \tag{5.7}$$

В результате, мы получили выражения (5.8) для неизвестных потоков дуг

для графа $\overline{G}$:

$$x_{1,5} - \frac{p_{1,5}}{p_{1,2}} x_{1,2} = 0,$$

$$x_{4,3} - \frac{p_{4,3}}{p_{4,8}} x_{4,8} = 0, \tag{5.8}$$

$$x_{4,7} - \frac{p_{4,7}}{p_{4,8}} x_{4,8} = 0.$$

Заметим, что для узлов $\{5, 8\} \subset I \setminus M^*$ невозможно получить дополнительные уравнения типа (1.5), так как для каждого из этих узлов есть только одна исходящая дуга.

Объединим (5.6) и (5.7). Таким образом, дополнительные уравнения типа (1.5) для графа $\overline{G}$ (см. рисунок 5.3) являются (5.8).

Часть неизвестных системы (5.5), (5.8) составляет потоки исходящих дуг для ребер из узлов множества $I \setminus I^* = \{1, 4, 5, 8\}$:

$$x_{1,2},\ \ x_{1,5},\ \ x_{4,3},\ \ x_{4,7},\ \ x_{4,8},\ \ x_{5,1},\ \ x_{8,4}.$$

Оставшаяся часть неизвестных системы (5.5), (5.8) определяет внешние потоки $x_i,\ i \in \overline{I}^*$, где $\overline{I}^* = \{2, 3, 7\}$:x_2, x_3, x_7.

Новый граф $\overline{G} = (\overline{I}, \overline{U})$ (см. рисунок 5.3) где $\overline{I}$ = $\{1, 2, 3, 4, 5, 7, 8\}$, $\overline{U}$ = $\{(1, 2), (1, 5), (4, 3), (4, 7), (4, 8), (5, 1), (8, 4)\}$, $\overline{I}^*$ = $\{2, 3, 7\}$ состоит из двух компонент связности: $\overline{G}_m = (\overline{I}_m, \overline{U}_m),\ m = 1, 2.$

Компонента связности $\overline{G}_1 = (\overline{I}_1, \overline{U}_1)$ состоит из узлов $\overline{I}_1$ = $\{1, 2, 5\}$, дуг $\overline{U}_1$ = $\{(1, 2), (1, 5), (5, 1)\}$ и узлов с внешними потоками $\overline{I}_1^*$ = $\{2\}$. Компонента связности $\overline{G}_2 = (\overline{I}_2, \overline{U}_2)$ состоит из узлов $\overline{I}_2$ = $\{3, 4, 7, 8\}$, дуг $\overline{U}_2$ = $\{(4, 3), (4, 7), (4, 8), (8, 4)\}$ и узлов с внешними потоками $\overline{I}_2^*$ где $\overline{I}_2^*$ = $\{3, 7\}$.

Выберем опору $R = \{U_R, I_R^*\}$ графа $\overline{G}$ для системы (5.5) (см. рисунок 5.4): U_R = $\{(1, 2), (1, 5), (4, 3), (4, 8)\}$, $I_R^* = \{2, 3, 7\}$.

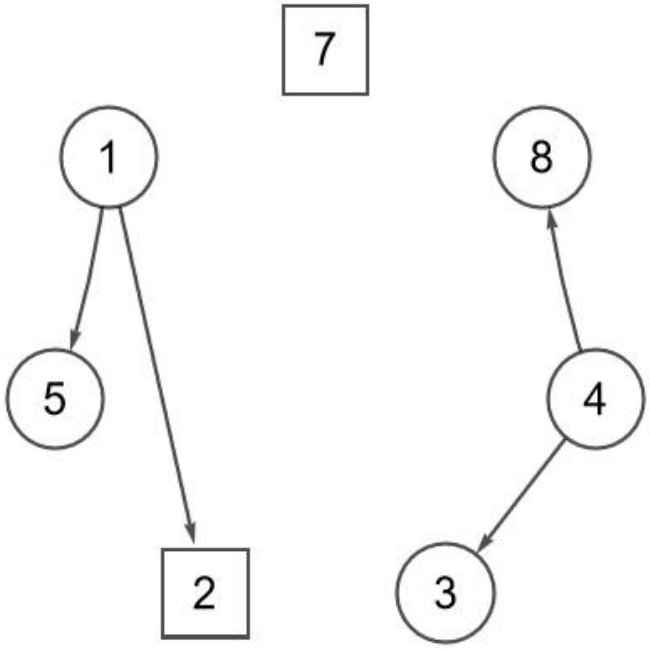

Рисунок 5.4 – Опора R графа $\overline{G}$ для системы (5.5)

После того, как опора $R = \{U_R, I_R^*\}$ графа $\overline{G}$ для системы (5.5) выбрана (см. рисунок (5.4)), мы определяем, какие структуры могут быть получены после добавления одного не опорного элемента из $\overline{U}\backslash U_R$ или $\overline{I}^*\backslash I_R^*$ в опору R. Мы строим систему характеристических векторов (базис пространства решений) однородной системы, порожденной системой (5.5). Векторы $\delta(4,7), \delta(5,1),$ $\delta(8,4)$, связанные дугами $(4,7), (5,1), (8,4)$, соответственно, образуют систему характеристических векторов. Вычислим характеристические векторы $\delta(4,7),$ $\delta(5,1), \delta(8,4)$. Вычисляем характеристический вектор $\delta(4,7) = (\delta_{i,j}^{4,7},\ (i,j) \in \overline{U};\ \delta_i^{4,7},\ i \in \overline{I}^*)$, связанный дугой $(4,7)$:

$$\delta(4,7) = (\delta_{1,2}^{4,7} \to 0, \delta_{1,5}^{4,7} \to 0, \delta_{4,3}^{4,7} \to -1, \delta_{4,7}^{4,7} \to 1,$$

$$\delta_{4,8}^{4,7} \to 0, \delta_{8,4}^{4,7} \to 0, \delta_{5,1}^{4,7} \to 0, \delta_2^{4,7} \to 0, \delta_3^{4,7} \to 1, \delta_7^{4,7} \to -1)$$

Вычисляем характеристические векторы, $\delta(5,1)$ сваязанные дугами $(5,1)$:

$$\delta(5,1) = (\delta_{1,2}^{5,1} \to 0, \delta_{1,5}^{5,1} \to 1, \delta_{4,3}^{5,1} \to 0, \delta_{4,7}^{5,1} \to 0,$$

$$\delta_{4,8}^{5,1} \to 0, \delta_{8,4}^{5,1} \to 0, \delta_{5,1}^{5,1} \to 1, \delta_2^{5,1} \to 0, \delta_3^{4,7} \to 0, \delta_7^{4,7} \to 0)$$

Вычисляем характеристические векторы, $\delta(8,4)$ связанные дугами $(8,4)$:

$$\delta(8,4) = (\delta_{1,2}^{8,4} \to 0, \delta_{1,5}^{8,4} \to 0, \delta_{4,3}^{8,4} \to 0, \delta_{4,7}^{8,4} \to 0,$$

$$\delta_{4,8}^{8,4} \to 1, \delta_{8,4}^{8,4} \to 1, \delta_{5,1}^{8,4} \to 0, \delta_2^{8,4} \to 0, \delta_3^{8,4} \to 0, \delta_7^{8,4} \to 0)$$

Перечислим дополнительные уравнения (5.8). Для каждого дополнительного уравнения с числом p мы вычисляем определители $\Lambda_{\tau\rho}^p$, $(\tau,\rho) \in \overline{U}\backslash U_R =$

$\{(4,7),(5,1),(8,4)\}$, $p=1,2,3$:

$$\Lambda^p_{\tau\rho}=\sum_{(i,j)\in U_R}\lambda^p_{ij}\delta^{\tau\rho}_{ij}+\lambda^p_{\tau\rho},\ p=\overline{1,q}$$

Формируем D. Матрица D состоит из определителей структур, не влекущих за собой никаких опорных элементов:

$$\Lambda^1_{4,7}=0,\Lambda^1_{5,1}=1,\Lambda^1_{8,4}=0,$$

$$\Lambda^2_{4,7}=-1,\Lambda^2_{5,1}=0,\Lambda^2_{8,4}=-\frac{p_{4,3}}{p_{4,8}},$$

$$\Lambda^3_{4,7}=1,\Lambda^3_{5,1}=0,\Lambda^3_{8,4}=-\frac{p_{4,7}}{p_{4,8}}.$$

Формируем множества $W=\{U_W,\ I^*_W\}$, $|W|=q$, $U_W\subseteq\overline{U}\setminus U_R$, $I^*_W\subseteq\overline{I}^*\setminus I^*_R$. Так как $I^*_W=\emptyset$, затем $W=U_W$, $U_W=\{(4,7),(5,1),(8,4)\}\}$. Перечисляем дуги множества U_W, где $t=t(\tau,\rho)$ – номер дуги $(\tau,\rho)\in U_W$ в введенной нумерации:

$$t(4,7)=1,\ \ t(5,1)=2,\ \ t(8,4)=3.$$

Формируем матрицу D:

$$D=\begin{pmatrix}\Lambda^1_{4,7} & \Lambda^1_{5,1} & \Lambda^1_{8,4}\\ \Lambda^2_{4,7} & \Lambda^2_{5,1} & \Lambda^2_{8,4}\\ \Lambda^3_{4,7} & \Lambda^3_{5,1} & \Lambda^3_{8,4}\end{pmatrix}$$

Таким образом, матрица D имеет вид:

$$D=\begin{pmatrix}0 & 1 & 0\\ -1 & 0 & -\dfrac{p_{4,3}}{p_{4,8}}\\ 1 & 0 & -\dfrac{p_{4,7}}{p_{4,8}}\end{pmatrix}$$

Поскольку справедливы следующие соотношения: $0<p_{i,j}\le 1,\ (i,j)\in U$, то определитель матрицы D не равен нулю:

$$\det D=-\frac{p_{4,3}}{p_{4,8}}-\frac{p_{4,7}}{p_{4,8}}\neq 0$$

Поэтому система (5.5), (5.8) является системой полного ранга. Кроме того, количество уравнений в системе (5.5), (5.8) равна рангу матрицы, следовательно, система (5.5), (5.8) имеет единственное решение для заданного множества $M=\{6\}$.

Построим единственное решение системы (5.5), (5.8). Применим алгоритмы декомпозицию для решения системы (5.5), (5.8). Для построения общего решения системы (5.5) необходимо построить любое частное решение $\tilde{x} = (\tilde{x}_{i,j}, (i,j) \in \overline{U},\, \tilde{x}_i,\, i \in \overline{I}^*)$ системы (5.5). Конкретное решение $\tilde{x}$ системы (5.5) строится в соответствии с правилами замечаний (2.4.1) из [13]. Несуществующие компоненты конкретного решения $\tilde{x}$ равны нулю $\tilde{x}_{\tau,\rho} = 0$, $(\tau,\rho) \in \overline{U} \backslash U_R,\ \tilde{x}_\gamma = 0, \gamma \in \overline{I}^* \backslash I_R^*$: $\tilde{x}_{4,7} = 0,\ \tilde{x}_{5,1} = 0,\ \tilde{x}_{8,4} = 0$. Использую свойства графов для опоры графа $\overline{G}$ для системы (5.5) опорные компоненты конкретного решение системы (5.5) мы найдем за время $O(n)$, $n = |\overline{I}|$. компоненты конкретного решение системы $\tilde{x}$ равны:

$$\tilde{x}_{1,2} = \frac{p_{2,1}}{p_{2,6}} f_{2,6}, \quad \tilde{x}_2 = f_{2,6} - f_{6,2},$$
$$\tilde{x}_{4,3} = \frac{p_{3,4}}{p_{3,6}} f_{3,6} + \frac{p_{7,4}}{p_{7,6}} f_{7,6},$$
$$\tilde{x}_3 = f_{3,6} - f_{6,3} - \frac{p_{7,4}}{p_{7,6}} f_{76}, \quad \tilde{x}_{1,5} = 0,$$
$$\tilde{x}_7 = \frac{p_{7,4}}{p_{7,6}} f_{7,6} + f_{7,6} - f_{6,7}, \quad \tilde{x}_{4,8} = 0.$$

Используя формулу (1.3.5) из [1], вычислим $A^1,\ A^2,\ A^3$:

$$A^1 = \frac{f_{2,6}\ p_{1,5} p_{2,1}}{p_{1,2}\ p_{2,6}},$$
$$A^2 = -\frac{p_{3,4}}{p_{3,6}} f_{3,6} - \frac{p_{7,4}}{p_{7,6}} f_{7,6},$$
$$A^3 = 0.$$

Так как матрица D несингулярна, применим формулу из [13] для нахождения компонент решения $x_W = (x_{\tau,\rho}, (\tau,\rho) \in U_W) = (x_{4,7},\ x_{5,1},\ x_{8,4})$ из системы в виде:

$$x_{5,1} = A^1,$$
$$-x_{4,7} - \frac{p_{4,3}}{p_{4,8}} x_{8,4} = A^2, \tag{5.9}$$
$$x_{4,7} - \frac{p_{4,7}}{p_{4,8}} x_{8,4} = A^3.$$

Компоненты вектора β – правая часть системы из [9] для рассматриваемого примера, имеют следующий вид:

$$\beta = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{pmatrix} = \begin{pmatrix} A^1 \\ A^2 \\ A^3 \end{pmatrix}$$

Вычисляем компоненты $x_W = (x_{4,7},\ x_{5,1},\ x_{8,4})$ вектора x_W из системы (5.9):

$$x_{5,1} = \frac{f_{2,6}\, p_{1,5}\, p_{2,1}}{p_{1,2}\, p_{2,6}},$$

$$x_{4,7} = \frac{p_{4,7}\,(f_{7,6}\, p_{3,6}\, p_{7,4} + f_{3,6}\, p_{3,4}\, p_{7,6})}{p_{3,6}\,(p_{4,3} + p_{4,7})\, p_{7,6}}, \tag{5.10}$$

$$x_{8,4} = \frac{p_{4,8}\,(f_{7,6}\, p_{3,6}\, p_{7,4} + f_{3,6}\, p_{3,4}\, p_{7,6})}{p_{3,6}\,(p_{4,3} + p_{4,7})\, p_{7,6}}.$$

Остальные компоненты искомого единственного решения системы (5.5), (5.8) находим по формулам из [15], используя сетевые свойства опоры $R = \{U_R, I_R^*\}$ графа $\overline{G}$ для системы (5.5). Имеем:

$$x_{1,2} = \frac{f_{2,6}\, p_{2,1}}{p_{2,6}},$$

$$x_2 = f_{2,6} - f_{6,2},$$

$$x_3 = f_{3,6} - f_{6,3} - \frac{f_{7,6}\, p_{7,4}}{p_{7,6}} + \frac{p_{4,7}\,(f_{7,6}\, p_{3,6}\, p_{7,4} + f_{3,6}\, p_{3,4}\, p_{7,6})}{p_{3,6}\,(p_{4,3} + p_{4,7})\, p_{7,6}},$$

$$x_{4,3} = \frac{p_{4,3}\,(f_{7,6}\, p_{3,6}\, p_{7,4} + f_{3,6}\, p_{3,4}\, p_{7,6})}{p_{3,6}\,(p_{4,3} + p_{4,7})\, p_{7,6}},$$

$$x_{1,5} = \frac{f_{2,6}\, p_{1,5}\, p_{2,1}}{p_{1,2}\, p_{2,6}},$$

$$x_7 = -f_{6,7} + \frac{-f_{3,6}\, p_{3,4}\, p_{4,7}\, p_{7,6} + f_{7,6}\, p_{3,6}(\, p_{4,7}\, p_{7,6} + p_{4,3}(p_{7,4} + p_{7,6}))}{p_{3,6}\,(p_{4,3} + p_{4,7})\, p_{7,6}},$$

$$x_{4,8} = \frac{p_{4,8}\,(f_{7,6}\, p_{3,6}\, p_{7,4} + f_{3,6}\, p_{3,4}\, p_{7,6})}{p_{3,6}\,(p_{4,3} + p_{4,7})\, p_{7,6}}.$$

Код программы, используемой для реализации аналитического решения, расположен в Приложении А.

5.3 Численный пример оценки потока на ненаблюдаемой части двунаправленной сети

Мы размещаем один датчик на узле 6. Значения дуговых потоков по всем входящим и исходящим дугам для каждого узла i из множества $M = \{6\}$ (наблюдаемых узлов) известны, а также известны значения внешних потоков $x_i = f_i,\ i \in M \bigcap I^*$.

Подставляем известные значения переменные:

$$f_{2,6} = 8,\ \ f_{3,6} = 3,\ \ f_{6,3} = 3,$$

$$f_{6,2} = 6,\ \ f_{6,7} = 5,\ \ f_{7,6} = 2,$$

$$f_6 = 1.$$

в систему уравнений (5.1). Удалим из графа $G = (I, U)$ множество дуг, на которых известны дуговые потоки:

$$(2,6),\ (3,6),\ (6,3),\ (6,2),\ (6,7),\ (7,6).$$

Помимо этого, удаляем из графа G множество вершин $i \in M$, $M = \{6\}$. Получаем граф G', который показан на рисунке 5.5.

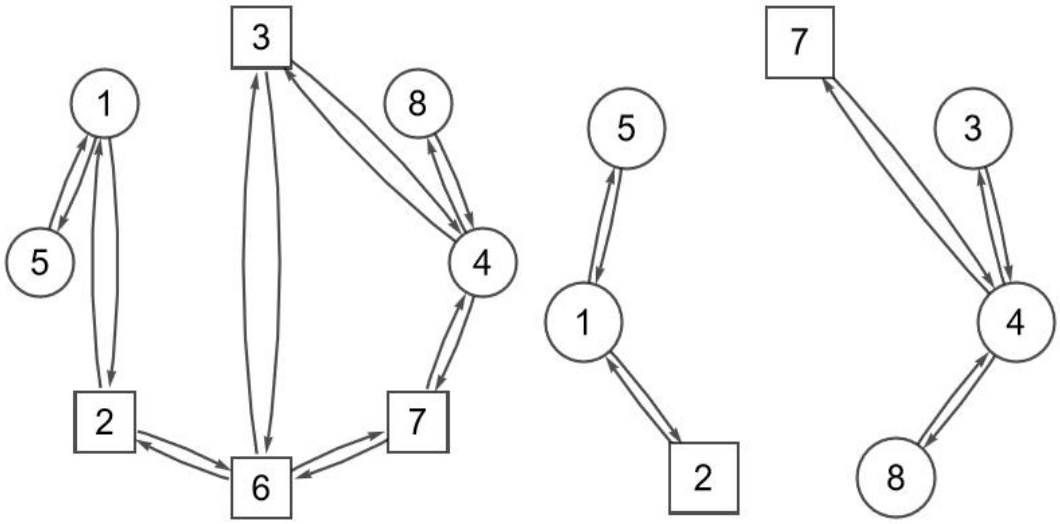

Рисунок 5.5 – Граф G (слева) и несвязный граф G' (справа)

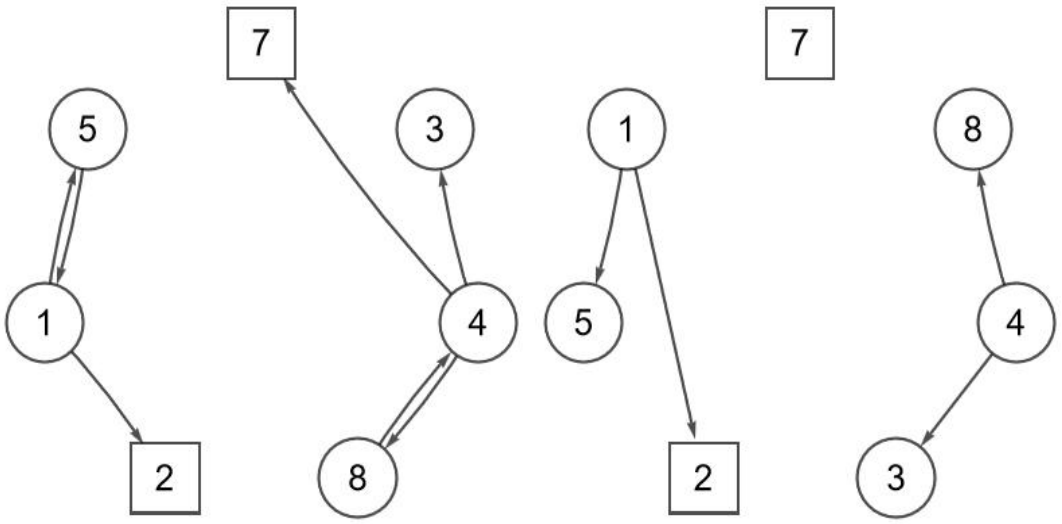

Рисунок 5.6 – Несвязный граф $\overline{G}$ (слева) и опора R графа $\overline{G}$(справа)

Остальные дуговые потоки для исходящих дуг из узлов множества $M^+ = I(CC(M)) \setminus M = \{2, 3, 7\}$, могут быть выражены из потоков исходящих дуг для M^+ следующим образом:

$$\begin{aligned} x_{2,1} &= \frac{p_{2,1}}{p_{2,6}} f_{2,6} = 8, \\ x_{3,4} &= \frac{p_{3,4}}{p_{3,6}} f_{3,6} = 9, \\ x_{7,4} &= \frac{p_{7,4}}{p_{7,6}} f_{7,6} = 1. \end{aligned} \tag{5.11}$$

Удалим из графа G' (см. рисунок 5.5) множество дуг, на которых известны известны потоки дуг согласно (5.11).

На рисунке 5.6 представлен граф $\overline{G} = (\overline{I},\ \overline{U})$ полученный путем удаления набор дуг, на которых известны дуговые потоки в соответствии с (5.11) из графа G' (см. рисунок 5.6).

Числовые значения $p_{i,j}, (i,j) \in U$ представлены в таблице 5.1.

Таблица 5.1 – Значения компонентов вектора $p = (p_{i,j},\ (i,j) \in U)$

(i,j)	(1,2)	(1,5)	(2,1)	(2,6)	(3,4)	(3,6)	(4,3)	(4,7)
$p_{i,j}$	$\frac{1}{3}$	$\frac{2}{3}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{3}{4}$	$\frac{1}{4}$	$\frac{5}{13}$	$\frac{4}{13}$

Окончание таблицы 5.1

(i,j)	(4,8)	(5,1)	(6,2)	(6,3)	(6,7)	(7,4)	(7,6)	(8,4)
$p_{i,j}$	$\frac{4}{13}$	1	$\frac{5}{10}$	$\frac{2}{10}$	$\frac{3}{10}$	$\frac{1}{3}$	$\frac{2}{3}$	1

Значения внешних потоков узлов из множества $I^* = \{2, 3, 6, 7\}$ равны:

$$x_2 = 2, \quad x_3 = 1, \quad x_6 = 1, \quad x_7 = -4.$$

Таблица 5.2 – Значение известных дуговых потоков $x = (x_{i,j}, (i,j) \in U)$

(i,j)	(1,2)	(1,5)	(2,1)	(2,6)	(3,4)	(3,6)	(4,3)	(4,7)
$x_{i,j}$	8	16	8	8	9	3	$\frac{50}{9}$	$\frac{40}{9}$

Окончание таблицы 5.2

(i,j)	(4,8)	(5,1)	(6,2)	(6,3)	(6,7)	(7,4)	(7,6)	(8,4)
$x_{i,j}$	$\frac{40}{9}$	16	6	3	5	1	2	$\frac{40}{9}$

В результате локализации одного датчика на узел 6 в графе G, который представлен на рисунке 5.5 мы имеем единственное решение системы (5.1). Дуговые потоки единственного решения системы (5.1) представлены в таблице 5.2.

ГЛАВА 6. ПРИМЕРЫЫ ПОСТРОЕНИЕ СУБОПТИМАЛЬНОГО РЕШЕНИЯ ЗАДАЧИ ОЦЕНКИ ПОТОКА НА НЕНАБЛЮДАЕМОЙ ЧАСТИ ДВУНАПРАВЛЕННОЙ СЕТИ

6.1 Пример 1

Рассмотрим первый пример нахождения субоптимально решения. Данный пример также опубликован в [16]. Рисунок 6.1 демонстрирует конечный связный двунаправленный граф $G = (I, U)$, где

$$I = \{1, 2, 3, 4, 5, 6, 7, 8\},$$

$$I^* = \{2, 3, 6, 7\},$$

$$U = \{(1, 2), (1, 5), (2, 1), (2, 6), (3, 4),$$
$$(3, 6), (4, 3), (4, 7), (4, 8), (5, 1),$$
$$(6, 2), (6, 3), (6, 7), (7, 4), (7, 6), (8, 4)\}.$$

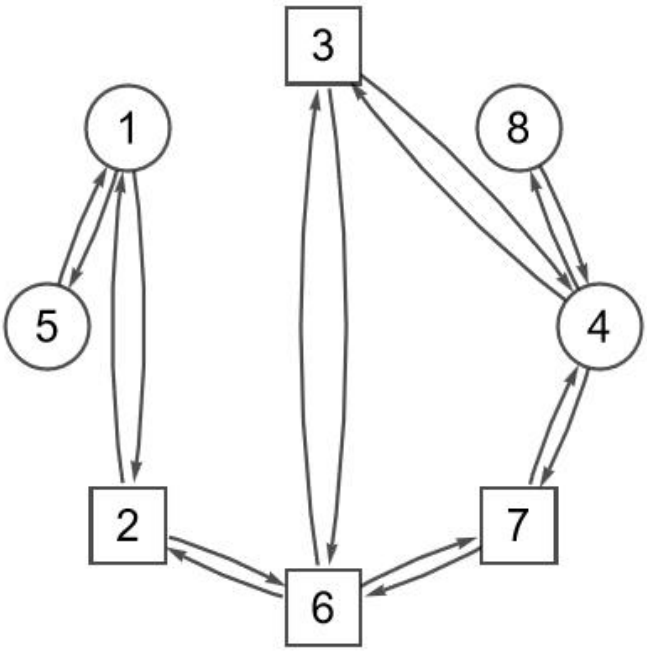

Рисунок 6.1 – Конечный связный двунаправленный граф G

Для графа G, изображенного на рисунке 6.1, рассмотрим систему линей-

ных алгебраических уравнений:

$$\begin{gathered} x_{1,2} + x_{1,5} - x_{2,1} - x_{5,1} = 0, \\ x_{2,1} + x_{2,6} - x_{1,2} - x_{6,2} = x_2, \\ x_{3,4} + x_{3,6} - x_{4,3} - x_{6,3} = x_3, \\ x_{4,3} + x_{4,7} + x_{4,8} - x_{3,4} - x_{7,4} - x_{8,4} = 0, \\ x_{5,1} - x_{1,5} = 0, \\ x_{6,2} + x_{6,3} + x_{6,7} - x_{2,6} - x_{3,6} - x_{7,6} = x_6, \\ x_{7,4} + x_{7,6} - x_{4,7} - x_{6,7} = x_7, \\ x_{8,4} - x_{4,8} = 0. \end{gathered} \tag{6.1}$$

Предположим, что множество обозреваемых узлов для графа G, изображенного на рисунке 6.1, равно $M = \{2, 3, 6, 7\}$. Обозначим разрез графа G относительно заданного множества узлов M через $CC(M)$. Построим разрез $CC(M)$ для графа G относительно множества $M = \{2, 3, 6, 7\}$.

Сформируем множества:

$$M^+ = I(CC(M)) \setminus M = \{1, 4\},$$

$$M^* = M \cup M^+ = \{2, 3, 6, 7\} \cup \{1, 4\} = \{1, 2, 3, 4, 6, 7\},$$

$$I \setminus M^* = \{5, 8\}.$$

Значения дуговых потоков по всем входящим и исходящим дугам для каждого узла i из множества $M = \{2, 3, 6, 7\}$ (контролируемый узел) известны:

$$\begin{gathered} x_{2,1} = f_{2,1}, \quad x_{1,2} = f_{1,2}, \\ x_{3,4} = f_{3,4}, \quad x_{3,4} = f_{3,4}, \\ x_{6,2} = f_{6,2}, \quad x_{2,6} = f_{2,6}, \\ x_{6,3} = f_{6,3}, \quad x_{3,6} = f_{3,6}, \\ x_{6,7} = f_{6,7}, \quad x_{7,6} = f_{7,6}, \\ x_{7,4} = f_{7,4}, \quad x_{4,7} = f_{4,7}. \end{gathered}$$

Помимо этого, нам также известны значения $x_i = f_i,\ i \in M \bigcap I^*$:

$$\begin{gathered} x_2 = f_2, \\ x_3 = f_3, \\ x_6 = f_6, \\ x_7 = f_7. \end{gathered}$$

Имеем:

$$\begin{aligned} x_{2,1} &= f_{2,1}, \quad x_{1,2} = f_{1,2}, \\ x_{3,4} &= f_{3,4}, \quad x_{3,4} = f_{3,4}, \\ x_{6,2} &= f_{6,2}, \quad x_{2,6} = f_{2,6}, \\ x_{6,3} &= f_{6,3}, \quad x_{3,6} = f_{3,6}, \\ x_{6,7} &= f_{6,7}, \quad x_{7,6} = f_{7,6}, \\ x_{7,4} &= f_{7,4}, \quad x_{4,7} = f_{4,7}, \\ x_2 &= f_2, \\ x_3 &= f_3, \\ x_6 &= f_6, \\ x_7 &= f_7. \end{aligned} \tag{6.2}$$

Подставим (6.2) в систему линейных уравнений (6.1). Удалим из графа $G = (I,\, U)$ множество дуг, на которых потоки по дугам известны в соответствии с (6.2). Также удалим из графа G множество вершин $i \in M$, $M = \{2, 3, 6, 7\}$. Мы получаем граф G', показанный на рисунке 6.2.

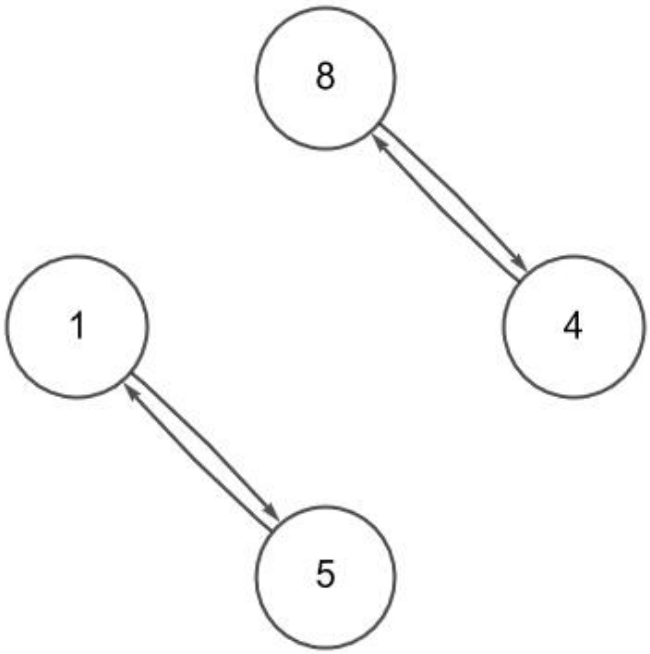

Рисунок 6.2 – Граф G'

Согласно [16], дуговые потоки для ребер, исходящих из узлов множества $M^+ = I(CC(M)) \setminus M = \{1, 4\}$, могут быть выражен из потока дуг, исходящих из множества M^+.

Получаем следующие уравнения:

$$\begin{aligned} x_{1,5} &= \frac{p_{1,5}}{p_{1,2}} x_{1,2}, \\ x_{1,2} &= f_{1,2}, \\ x_{4,8} &= \frac{p_{4,8}}{p_{4,3}} x_{4,3}, \\ x_{4,3} &= f_{4,3}. \end{aligned} \tag{6.3}$$

На рисунке 6.3 представлен граф $\overline{G} = (\overline{I},\, \overline{U})$ полученный путем удаления множества ребер, на которых дуговой поток известен в соответствии с (6.3) для графа G' (см. рисунок 6.2). Система (6.1) для графа $\overline{G}$ (см. рисунок 6.3) преобразуется к виду (6.4).

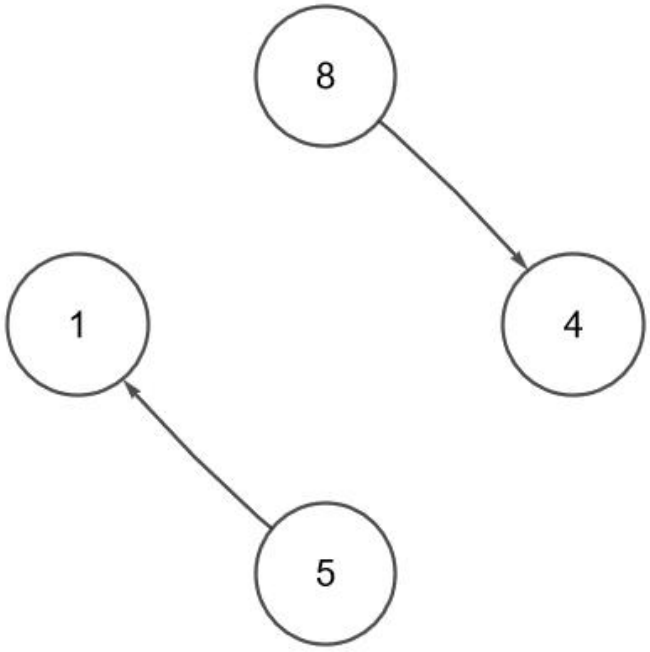

Рисунок 6.3 – Граф $\overline{G}$

Дополнительные уравнения отсутствуют, поскольку в узлах ненаблюдаемой части графа отсутствуют узлы, из которых исходит более 1 дуги.

$$
\begin{gathered}
f_{1,2} + \frac{p_{1,5}}{p_{1,2}} f_{1,2} - f_{2,1} - x_{5,1} = 0, \\
f_{2,1} + f_{2,6} - f_{1,2} - f_{6,2} = f_2, \\
f_{3,4} + f_{3,6} - f_{4,3} - f_{6,3} = f_3, \\
f_{4,3} + f_{4,7} + \frac{p_{4,8}}{p_{4,3}} f_{4,3} - f_{3,4} - f_{7,4} - x_{8,4} = 0, \\
x_{5,1} - \frac{p_{1,5}}{p_{1,2}} x_{1,2} = 0, \\
f_{6,2} + f_{6,3} + f_{6,7} - f_{2,6} - f_{3,6} - f_{7,6} = f_6, \\
f_{7,4} + f_{7,6} - f_{4,7} - f_{6,7} = f_7, \\
x_{8,4} - \frac{p_{4,8}}{p_{4,3}} x_{4,3} = 0.
\end{gathered}
\tag{6.4}
$$

В системе остается лишь 2 неизвестные переменные, и решение данной системы выглядит следующим образом:

$$
\begin{gathered}
x_{5,1} = x_{1,5} = \frac{p_{1,5}}{p_{1,2}} x_{1,2}, \\
x_{8,4} = x_{4,8} = \frac{p_{4,8}}{p_{4,3}} x_{4,3}.
\end{gathered}
\tag{6.5}
$$

6.2 Пример 2

Для второго примера был выбран граф $G = (I, U)$ с большим число вершин $|I|$ и ребер $|U|$. Такой выбор был сделан для демонстрации возможностей приведенной ранее программной реализации построения субоптимального решения задачи оценки потока на ненаблюдаемой части графа. Граф изображен на рисунке 6.4.

$$
\begin{gathered}
I = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15\}, \\
I^* = \{4, 5, 6, 11, 12, 13\}, \\
U = \{(1,2), (2,1), (2,3), (2,5), (3,2), (3,4), \\
(4,3), (4,5), (5,2), (5,4), (5,6), (6,5),
\end{gathered}
$$

$$(6,7),(7,6),(7,8),(8,7),(8,9),(8,11)$$
$$(9,2),(9,5),(9,8),(9,10),(9,13),(10,9),$$
$$(10,11),(11,8),(11,10),(11,12),(12,11),(12,13),$$
$$(13,9),(13,12),(13,14),(14,13),(14,15),(15,14),(15,1)\}.$$

Для данного графа формируем систему линейных алгебраических уравнений:

$$x_{1,2}+x_{1,15}-x_{2,1}-x_{15,1}=0,$$
$$-x_{1,2}+x_{2,1}+x_{2,3}+x_{2,5}+x_{2,9}-x_{3,2}-x_{5,2}-x_{9,2}=0,$$
$$-x_{2,3}+x_{3,2}+x_{3,4}-x_{4,3}=0,$$
$$-x_{3,4}+x_{4,3}+x_{4,5}-x_{5,4}=x_4,$$
$$-x_{2,5}-x_{4,5}+x_{5,2}+x_{5,4}+x_{5,6}+x_{5,9}-x_{6,5}-x_{9,5}=x_5,$$
$$-x_{5,6}+x_{6,5}+x_{6,7}-x_{7,6}=x_6,$$
$$-x_{6,7}+x_{7,6}+x_{7,8}-x_{8,7}=0,$$
$$-x_{7,8}+x_{8,7}+x_{8,9}+x_{8,11}-x_{9,8}-x_{11,8}=0,$$
$$-x_{2,9}-x_{5,9}-x_{8,9}+x_{9,2}+x_{9,5}+x_{9,8}+x_{9,10}+x_{9,13}-x_{10,9}-x_{13,9}=0,$$
$$-x_{9,10}+x_{10,9}+x_{10,11}-x_{11,10}=0,$$
$$-x_{8,11}-x_{10,11}+x_{11,8}+x_{11,10}+x_{11,12}-x_{12,11}=x_{11},$$
$$-x_{11,12}+x_{12,11}+x_{12,13}-x_{13,12}=x_{12},$$
$$-x_{9,13}-x_{12,13}+x_{13,9}+x_{13,12}+x_{13,14}-x_{14,13}=x_{13},$$
$$-x_{13,14}+x_{14,13}+x_{14,15}-x_{15,14}=0,$$
$$-x_{1,15}-x_{14,15}+x_{15,1}+x_{15,14}=0. \tag{6.6}$$

Множество обозреваемых узлов графа G равно $M=I^*=\{4,5,6,11,12,13\}$. Как и в примере 1, строим разрез $CC(M)$ графа G относительно множества M.

Дополнительные множества имеют вид:

$$M^+=I(CC(M))\setminus M=\{2,3,7,8,9,10,14\},$$
$$M^*=M\cup M^+=\{4,5,6,11,12,13\}\cup\{2,3,7,8,9,10,14\}=$$
$$\{2,3,4,5,6,7,8,9,10,11,12,13,14\},$$
$$I\setminus M^*=\{1,15\}.$$

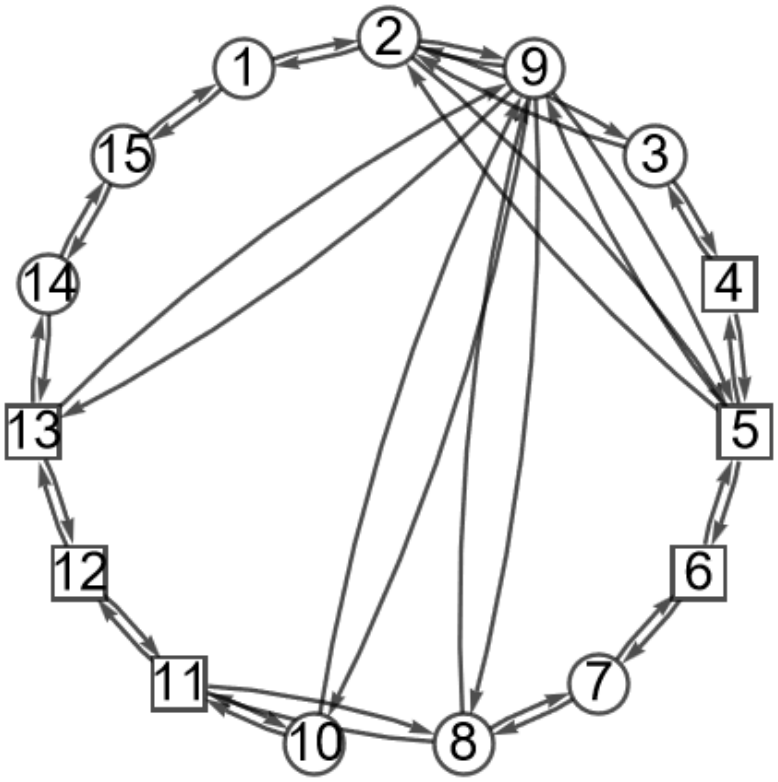

Рисунок 6.4 – Конечный связный двунаправленный граф G

Для каждого просматриваемого узла из множества $M = \{4, 5, 6, 11, 12, 13\}$ известны значения духовых потоков для каждой входящей и исходящей дуги. Также известны значения $x_i = f_i,\ i \in M \bigcap I^*$: Приведем все извенстные данные ниже:

$$
\begin{aligned}
x_{4,3} = f_{4,3}, \quad & x_{3,4} = f_{3,4}, \\
x_{4,5} = f_{4,5}, \quad & x_{5,4} = f_{5,4}, \\
x_{5,2} = f_{5,2}, \quad & x_{2,5} = f_{2,5}, \\
x_{5,6} = f_{5,6}, \quad & x_{6,5} = f_{6,5}, \\
x_{5,9} = f_{5,9}, \quad & x_{9,5} = f_{9,5}, \\
x_{6,7} = f_{6,7}, \quad & x_{7,6} = f_{7,6}, \\
x_{11,8} = f_{11,8}, \quad & x_{8,11} = f_{8,11}, \\
x_{11,10} = f_{11,10}, \quad & x_{10,11} = f_{10,11}. \\
x_{11,12} = f_{11,12}, \quad & x_{12,11} = f_{12,11}, \\
x_{12,13} = f_{12,13}, \quad & x_{13,12} = f_{13,12}, \\
x_{13,9} = f_{13,9}, \quad & x_{9,13} = f_{9,13}, \\
x_{13,14} = f_{13,14}, \quad & x_{14,13} = f_{14,13},
\end{aligned}
\tag{6.7}
$$

$$
\begin{gathered}
x_4 = f_4, \\
x_5 = f_5, \\
x_6 = f_6, \\
x_{11} = f_{11}, \\
x_{12} = f_{12}, \\
x_{13} = f_{13}.
\end{gathered}
\tag{6.8}
$$

Подставим системы (6.7) и (6.8) в систему (6.6). Получим:

$$
\begin{gathered}
x_{1,2} + x_{1,15} - f_{2,1} - x_{15,1} = 0, \\
-x_{1,2} + x_{2,1} + x_{2,3} + f_{2,5} + x_{2,9} - x_{3,2} - f_{5,2} - x_{9,2} = 0, \\
-x_{2,3} + x_{3,2} + f_{3,4} - f_{4,3} = 0, \\
-f_{3,4} + f_{4,3} + f_{4,5} - f_{5,4} = f_4, \\
-f_{2,5} - f_{4,5} + f_{5,2} + f_{5,4} + f_{5,6} + f_{5,9} - f_{6,5} - f_{9,5} = f_5, \\
-f_{5,6} + f_{6,5} + f_{6,7} - f_{7,6} = f_6, \\
-f_{6,7} + f_{7,6} + x_{7,8} - x_{8,7} = 0, \\
-x_{7,8} + x_{8,7} + x_{8,9} + f_{8,11} - x_{9,8} - f_{11,8} = 0, \\
-x_{2,9} - f_{5,9} - x_{8,9} + x_{9,2} + f_{9,5} + x_{9,8} + x_{9,10} + f_{9,13} - x_{10,9} - f_{13,9} = 0, \\
-x_{9,10} + x_{10,9} + f_{10,11} - f_{11,10} = 0, \\
-f_{8,11} - f_{10,11} + f_{11,8} + f_{11,10} + f_{11,12} - f_{12,11} = f_{11}, \\
-f_{11,12} + f_{12,11} + f_{12,13} - f_{13,12} = f_{12}, \\
-f_{9,13} - f_{12,13} + f_{13,9} + f_{13,12} + f_{13,14} - f_{14,13} = f_{13}, \\
-f_{13,14} + f_{14,13} + x_{14,15} - x_{15,14} = 0, \\
-x_{1,15} - x_{14,15} + x_{15,1} + x_{15,14} = 0.
\end{gathered}
\tag{6.9}
$$

Удалим из графа $G = (I,\, U)$ множество дуг с известными дуговыми потоками в соответствии с (6.9) и множество вершин $M = \{2, 3, 6, 7\}$. В результате получаем граф G', изображенный на рисунке 6.5.

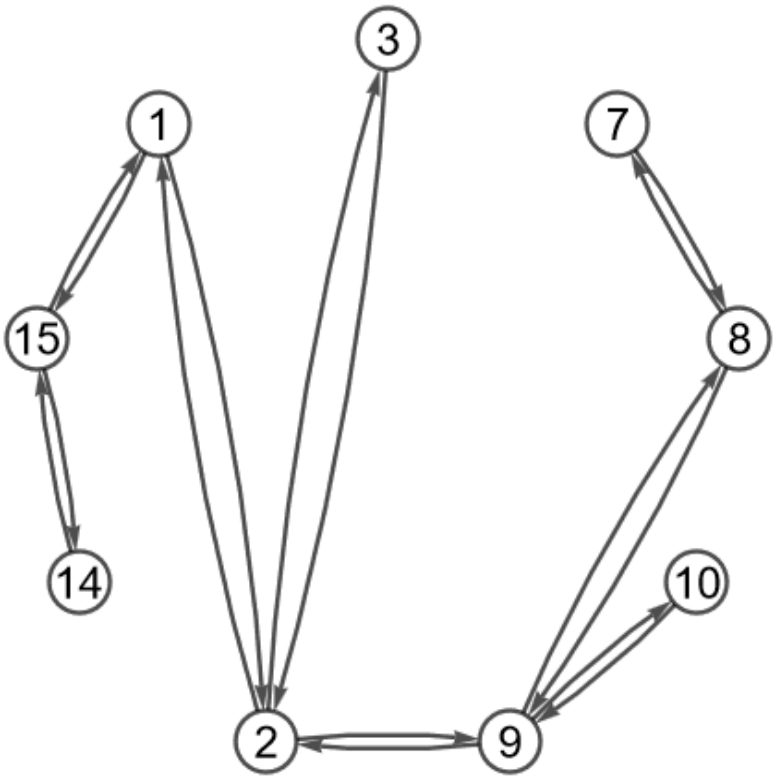

Рисунок 6.5 – Граф G

Для каждой вершины $i \in M^+, M^+ = 2, 3, 7, 8, 9, 10, 14$ все исходящие дуги выражаются при помощи дополнительных уравнений, так как число исходящих дуг для каждой вершины больше 1. Имеем:

$$\begin{aligned}
x_{2,1} &= \frac{p_{2,1}}{p_{2,5}} f_{2,5}, \quad x_{2,3} = \frac{p_{2,3}}{p_{2,5}} f_{2,5}, \\
x_{2,9} &= \frac{p_{2,9}}{p_{2,5}} f_{2,5}, \quad x_{3,2} = \frac{p_{3,2}}{p_{3,4}} f_{3,4}, \\
x_{7,8} &= \frac{p_{7,8}}{p_{7,6}} f_{7,6}, \quad x_{8,7} = \frac{p_{8,7}}{p_{8,11}} f_{8,11}, \\
x_{8,9} &= \frac{p_{8,9}}{p_{8,11}} f_{8,11}, \quad x_{9,2} = \frac{p_{9,2}}{p_{9,5}} f_{9,5}, \\
x_{9,8} &= \frac{p_{9,8}}{p_{9,5}} f_{9,5}, \quad x_{9,10} = \frac{p_{9,10}}{p_{9,5}} f_{9,5}, \\
x_{10,9} &= \frac{p_{10,9}}{p_{10,11}} f_{10,11}, \quad x_{14,15} = \frac{p_{14,15}}{p_{14,13}} f_{14,13}.
\end{aligned} \tag{6.10}$$

Подставляем выражения из (6.10) в систему (6.9). В результате получаем систему уравнений (6.11):

$$
\begin{gathered}
x_{1,2} + x_{1,15} - f_{2,1} - x_{15,1} = 0, \\
-x_{1,2} + \frac{p_{2,1}}{p_{2,5}} f_{2,5} + \frac{p_{2,3}}{p_{2,5}} f_{2,5} + f_{2,5} + \frac{p_{2,9}}{p_{2,5}} f_{2,5} - \\
\frac{p_{3,2}}{p_{3,4}} f_{3,4} - f_{5,2} - \frac{p_{9,2}}{p_{9,5}} f_{9,5} = 0, \\
-\frac{p_{2,3}}{p_{2,5}} f_{2,5} + \frac{p_{3,2}}{p_{3,4}} f_{3,4} + f_{3,4} - f_{4,3} = 0, \\
-f_{3,4} + f_{4,3} + f_{4,5} - f_{5,4} = f_4, \\
-f_{2,5} - f_{4,5} + f_{5,2} + f_{5,4} + f_{5,6} + f_{5,9} - f_{6,5} - f_{9,5} = f_5, \\
-f_{5,6} + f_{6,5} + f_{6,7} - f_{7,6} = f_6, \\
-f_{6,7} + f_{7,6} + \frac{p_{7,8}}{p_{7,6}} f_{7,6} + x_{8,7} - \frac{p_{8,7}}{p_{8,11}} f_{8,11} = 0, \\
-\frac{p_{7,8}}{p_{7,6}} f_{7,6} + \frac{p_{8,7}}{p_{8,11}} f_{8,11} + \frac{p_{8,9}}{p_{8,11}} f_{8,11} + f_{8,11} - \frac{p_{9,8}}{p_{9,5}} f_{9,5} - f_{11,8} = 0, \\
-\frac{p_{2,9}}{p_{2,5}} f_{2,5} - f_{5,9} - \frac{p_{8,9}}{p_{8,11}} f_{8,11} + \frac{p_{9,2}}{p_{9,5}} f_{9,5} + f_{9,5} + \frac{p_{9,8}}{p_{9,5}} f_{9,5} + \\
\frac{p_{9,10}}{p_{9,5}} f_{9,5} + f_{9,13} - \frac{p_{10,9}}{p_{10,11}} f_{10,11} - f_{13,9} = 0, \\
-\frac{p_{9,10}}{p_{9,5}} f_{9,5} + \frac{p_{10,9}}{p_{10,11}} f_{10,11} + f_{10,11} - f_{11,10} = 0, \\
-f_{8,11} - f_{10,11} + f_{11,8} + f_{11,10} + f_{11,12} - f_{12,11} = f_{11} \\
-f_{11,12} + f_{12,11} + f_{12,13} - f_{13,12} = f_{12}, \\
-f_{9,13} - f_{12,13} + f_{13,9} + f_{13,12} + f_{13,14} - f_{14,13} = f_{13}, \\
-f_{13,14} + f_{14,13} + \frac{p_{14,15}}{p_{14,13}} f_{14,13} - x_{15,14} = 0, \\
-x_{1,15} - \frac{p_{14,15}}{p_{14,13}} f_{14,13} + x_{15,1} + x_{15,14} = 0.
\end{gathered}
\tag{6.11}
$$

Удалим из графа G' множество ребер в соответствии с (6.10). В результате получим граф $\overline{G}$, соответствующий системе (6.11) и изображенный на рисунке 6.6.

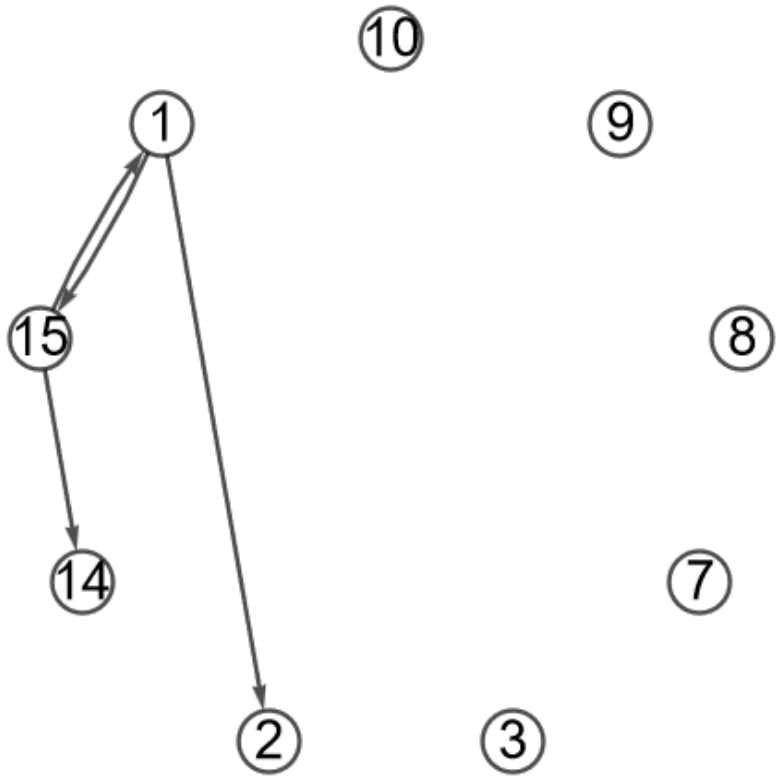

Рисунок 6.6 – Граф $\overline{G}$

Таким образом, получаем 4 неизвестные, относительно которых в дальнейшем будем решать систему (6.11): $x_{1,2}, x_{1,15}, x_{15,1}, x_{15,1}$. Из системы уравнений, полученной после полного сбора информации, можем найти $x_{1,2}$ и $x_{15,14}$ путем решения второго и четырнадцатого уравнений. Получим:

$$\begin{aligned} x_{1,2} &= \frac{p_{2,1}f_{2,5} + p_{2,5}(f_{13,14} - f_{14,13})}{p_{2,5}}, \\ x_{15,14} &= \frac{p_{14,13}(f_{14,13} - f_{13,14}) + p_{14,15}f_{14,13}}{p_{14,13}}. \end{aligned} \tag{6.12}$$

Так как все вершины множества

$$I \setminus M^* = \{1, 15\}$$

имеют больше одной исходящей дуги, то для них справедливы дополнительные уравнения:

$$\begin{aligned} x_{1,15} &= \frac{p_{1,15}}{p_{1,2}} x_{1,2}, \\ x_{15,1} &= \frac{p_{15,1}}{p_{15,14}} x_{15,14}. \end{aligned} \tag{6.13}$$

Подставляя решения (6.12) в дополнительные уравнения (6.13), получаем дуговые потоки $x_{1,15}$ и $x_{15,1}$:

$$x_{1,15} = \frac{p_{1,15}(p_{2,1}f_{2,5} + p_{2,5}(f_{13,14} - f_{14,13})}{p_{1,2}p_{2,5}}$$

$$x_{15,1} = \frac{p_{15,1}(p_{14,13}(f_{14,13} - f_{13,14}) + p_{14,15}f_{14,13})}{p_{14,13}p_{15,14}}$$

В результате получаем решение системы уравнений (6.11). Решение системы имеет вид:

$$\begin{aligned}
x_{1,2} &= \frac{p_{2,1}f_{2,5} + p_{2,5}(f_{13,14} - f_{14,13})}{p_{2,5}}, \\
x_{15,14} &= \frac{p_{14,13}(f_{14,13} - f_{13,14}) + p_{14,15}f_{14,13}}{p_{14,13}}, \\
x_{1,15} &= \frac{p_{1,15}(p_{2,1}f_{2,5} + p_{2,5}(f_{13,14} - f_{14,13})}{p_{1,2}p_{2,5}}, \\
x_{15,1} &= \frac{p_{15,1}(p_{14,13}(f_{14,13} - f_{13,14}) + p_{14,15}f_{14,13})}{p_{14,13}p_{15,14}}.
\end{aligned} \tag{6.14}$$

ЗАКЛЮЧЕНИЕ

В ходе работы:

1. Сформулирована задача поиска оптимального решения задачи расположения сенсоров для оценки потока на ненаблюдаемой части сети
2. Представлена математическая модель задачи оценки потоков в двунаправленном графе
3. Построены символьный и численный примеры построения оптимального решения
4. Реализован алгоритм визуализации двунаправленного связного графа в системе Wolfram Mathematica
5. Построено аналитическое решение разряженной системы линейных уравнений для случая установки сенсора в 1 узел
6. Построено численный пример оценки потока на ненаблюдаемой части двунаправленной сети
7. Определена свойста субоптимальных решений задачи расположения сенсоров для оценки потока
8. Реализована программа генерации графа и данных о внешних потоках сети в системе Wolfram Mathematica
9. Выполнена визуализации двунаправленного графа и системы трафика для поиска субоптимального решения
10. Реализован алгоритм сбора информации и формирование ненаблюдаемой части графа (сети)
11. Реализация алгоритмов вычисления дуговых потоков ненаблюдаемой части сети в Wolfram Mathematica
12. Реализация алгоритмических, структурных и технологических решений задачи оценки однородного потока в двунаправленной сети в системе Wolfram Mathematica
13. Дипломная работа выполнена в системе компьютерной верстки LaTex

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Адельсон-Вельский, Г. М. Потоковые алгоритмы / Г. М. Адельсон-Вельский, Е. А. Диниц, А. В. Карзанов - Москва, 1975. - 119 с.
2. Габасов, Р. Методы линейного программирования. Ч. 1: Общие задачи / Р. Габасов, Ф. М. Кириллова - Минск, 1977. - 174 с.
3. Габасов, Р. Методы линейного программирования. Ч. 2: Транспортные задачи / Р. Габасов, Ф. М. Кириллова - Минск, 1978. - 239 с.
4. Гантмахер, Ф. Р. Теория матриц / Ф. Р. Гантмахер. - Москва, Наука, 1988. - 581 с.
5. Иванчев Д. Сетевая оптимизация / Д. Иванчев - София, 2002.
6. Котов, В.М. Алгоритмы и структуры данных: учеб. пособие / В.М. Котов, Е.П. Соболевская, А.А. Толстиков. – Минск: БГУ, 2011. – 267 с.
7. Пилипчук, Л.А. Идентификация сенсорной конфигурации и управления потоками / Л.А. Пилипчук, А.С. Пилипчук, Е.Н. Полячок, А.И. Фаразей – Минск : БГУ, 2018.
8. Пилипчук, Л. А. Разреженные недоопределенные системы линейных алгебраических уравнений / Л. А. Пилипчук. – Минск : БГУ, 2012. – 260 с.
9. Пилипчук, Л.А. Реализация методов декомпозиции в задаче оценки потока на ненаблюдаемой части двунаправленной сети / Л. А. Пилипчук, М. П. Романчук // АМАДЕ-2021, 2021 - с. 65-77
10. Романовский, И. В Субоптимальное решение / И.В. Романовский - Петрозаводск, Издательство Петрозаводского университета. - 1998.
11. Фидлер М, Недома Й, Рамик Я, Рон И, Циммерманн К. Задачи линейной оптимизации с неточными данными / М. Фидлер, Й. Недома, Я. Рамик, И. Рон, К. Циммерманн - Москва: Институт компьютерных исследований, 2008. - 288 с.
12. Bianco L. A network based model for traffic sensor location with implication in O/D matrix estimates / L. Bianco, G. Confessore, M. Gentili // Transportation Science. Vol. 35, No. 1. - 2001. - P. 50 - 60
13. Bianco L. Aspects of the Sensor Location Problem / L. Bianco, G. Confessore, M. Gentili// Annals of Operation Research, 2006. № 144 (1). P. 201–234.
14. Bianco L. Locating sensors to observe network arc flows: exact and heuristic approaches. Computers and Operation Research. / L. Bianco, C. Cerrone, R. Cerulli, M. Gentili- 2014
15. Pilipchuk, L.A. Sparse linear systems and their applications / L.A. Pilipchuk. – Minsk : BSU, 2013. – 235 p.
16. Ramanouski Y. V. About suboptimal solutions to the problem of identifying of special programmable devices (sensors) and flows control / Ramanouski Y. V., Pilipchuk L. A., Biarozka I. S. // ITS-2024, 2024 - P. 135-135.

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ А

Листинг программы (оптимальное решение)

```
Off[Syntax::stresc]
SetDirectory[DirectoryName[ToFileName["FileName"/.
NotebookInformation[EvaluationNotebook[]]]]];
{II, SS, U} = Select[ReadList["8-8.txt"], #=!=Null&];
I2 = Complement[II, SS];
Print["I=", II];
Print["S=", SS];
Print["I\S=", I2];
Print["U=", U];
nI = Length[II];
U = Join[U, U/.{x_, y_} :→ {y, x}];
U = Cases[Tally[U], {{x_, y_}, _}:>{x, y}]
G = Graph[II, Cases[Tally[U], {{x_, y_}, _}:>{x, y}], DirectedEdges->True];
Print["The Graph looks as following"];
Graph[II, U, VertexLabels->Placed["Name", Center], VertexSize->Large,
VertexLabelStyle->Directive[Black, 14],
  VertexStyle->Directive[EdgeForm[Thick], White],
VertexShapeFunction->{Alternatives@@SS->"Square"},
    EdgeStyle->Directive[Black, Thick], GraphLayout->"CircularEmbedding",
DirectedEdges->True]
For[i = 1, i<=nI, ,
For[j = 1, j<=Length[I2], j++,
If [Length [qq[[j]]] == 0, Continue[]] ;
i++;
n = First [qq[[j]]] ;
qq[[j]] = Rest [qq[[j]]] ;
Var[base_, idx_]:=Subscript@@Prepend[idx, base];
Coeff[MM_, n_]:=Block[{out, j, v},
If[MemberQ[MM, n], Return[{}]];
out = Cases[U, {n, _}];
If[Length[out] ≤ 1, Return[{}]];
j = Cases[out, {n, x_}/;MemberQ[MM, x]];
v = If[j == {}, First[out], First[j]];
```

```
Return [(Var[x, #]==Var[p,#]/Var[p,v] Var[x, v]) &/@Complement[out, j, {v}]]
];
Generate[MM_]:=Block[{eq0, eq1, eq2, syst, MMM, IM, SM},
IM = Complement[II, MM]; (* I\M *)
MMM = Union@@(Join@@Cases[U, {#, _}]&/@MM); (*M^+*)
SM = Complement[SS, MM]; (* S\M *)
eq0 = (Plus@@(Var[x, #]&/@Cases[U, {#, _}]) − Plus
@@(Var[x, #]&/@Cases[U, {_, #}]) == If[Cases[SS, #] == {}, 0,
Subscript[S, #]])&/@IM;
eq1 = Var[x, #] == Var[f, #]&/@Union@@(Cases[U, {#, _}|{_, #}]&/@MM);
eq2 = Join@@(Coeff[MM, #]&)/@IM;
syst = Join[eq0, eq1, eq2, eq3];
vars = Union@Cases[syst, Subscript[x, _, _]|Subscript[S, _], Infinity];
Return[{syst, vars}];
Print["Results"];
IM = Complement[II, Solution];
MMM = Union@@(Join@@Cases[U, {#, _}]&/@Solution);
SM = Complement[SS, Solution];
Print["I=", II, ",
U = ", U, ", I\M=",
IM, ", S=", SS, "S\M=", SM, ", M="M*=", MMM,", |S\M|=",
Length[SM], ",|M*|=", .
Length[MMM], ", |M|=",
Length[Solution], ", |S\M| == |M*|-|M| ? ",
Solution, ", M^+=", Complement[MMM, Solution], ",
Length[SM] == Length[MMM] − Length[Solution]];
Print[syst];
solve = Solve@@syst;
Print[MatrixForm[Transpose[solve]]];
Print[Simplify[syst[[1]]/.solve]];
```

Листинг программы (субоптимальное решение)

```
ClearAll["Global*"]
SetDirectory[NotebookDirectory[]];
readGraph2[file_, dir_]:=Module[{
   fn = FileNameJoin[{dir, file}],
   stream, imod, umod, u, b
   },
   stream = OpenRead[fn];
   imod = Read[stream, {Word, Number}][[2]];
   umod = Read[stream, {Word, Number}][[2]];
   u = ({#[[1]]#[[2]], #[[2]]#[[1]]} &/@ReadList[stream, Expression, umod]) //Flatten;
b = ConstantArray[0, imod];
   (b[[Read[StringToStream[StringTake[#1, {5, -3}]], Number]]] = #2)&@@@
ReadList[stream, {Word, Expression}, imod];
{Graph[u, VertexSize->Medium, VertexLabels → Placed["Name", Center],
VertexStyle → Directive[White], VertexShapeFunction → {xx_ :→
If[SameQ[b[[xx]], x], "Square", "Circle"]}, VertexLabelStyle->Directive[Black, 24],
GraphLayout->"CircularEmbedding"], b}]
forma[ff_]:= ((ff/. {ξ_u_v_ → ξu,v}) //TableForm)
BuildSpanningTree[g_]:=Module[{s = {}, root = tRoot},
tPred = ConstantArray[0, Length[VertexList[g]]];
listUt = {};
tDepth = ConstantArray[0, Length[VertexList[g]]];
tDir = ConstantArray[0, Length[VertexList[g]]];
tDinast = {};
DepthFirstScan[UndirectedGraph[g], root, {"FrontierEdge"->
Function[e, {AppendTo[s, e[[1]]->e[[2]]],
tPred[[e[[2]]]] = e[[1]],
tDepth[[e[[2]]]] = 1 + tDepth[[e[[1]]]],
For[i = 1, i<=Length[arcsn], i++,
{y, z} = List@@arcsn[[i]];
If[e[[1]]==y&&e[[2]]==z,
AppendTo[listUt, y->z]; tDir[[e[[2]]]] = 1; ];
If[e[[2]]==y&&e[[1]]==z,
AppendTo[listUt, y->z];
tDir[[e[[2]]]] = −1]; ]}],
"PrevisitVertex"->Function[u, AppendTo[tDinast, u]]}];
Return[s];
```

```
];
tRoot = 4;
arcsn = {};
arclist = {};
fileContent = Import[.\\input1.txt]
elements = StringSplit[fileContent];
vertexCount = ToExpression[elements[[2]]];
edgeCount = ToExpression[elements[[4]]];
For[i = 0, i < edgeCount, i++,
{z, y} = ToExpression[elements[[i + 5]]];
AppendTo[arcsn, z → y];
AppendTo[arcsn, y → z];
AppendTo[arclist, {z, y}]; ]
edgeCount = edgeCount * 2;
{g, b} = readGraph2["input1.txt", NotebookDirectory[]];
vert = VertexList[g];
newVert = Sort[vert];
gr = Graph[newVert, EdgeList[g]];
b = MapIndexed [#1/.x → x_{#2[[1]]}&, b] [[#]]&/@VertexList[gr];
GraphPlot[g, EdgeStyle → Directive[Black, Thick],
VertexLabels->Placed["Name", Center],
VertexSize->Large, VertexStyle → Directive[EdgeForm[Thick], White],
MultiedgeStyle → .07]
balanceEqs = ((Total [x_#&/@EdgeList[g, #_]] − Total [x_#&/@EdgeList[g, _#]]))
== b[[#]]&/@VertexList[gr];
TableForm[balanceEqs]
tree = BuildSpanningTree[g];
Print["Root tree"]
TreeGraph[tree, DirectedEdges → True, VertexLabels → Placed["Name", Center],
VertexSize → 0.3, VertexLabelStyle → 15,
EdgeShapeFunction->{{"Arrow", "ArrowSize" → 0.06}},
VertexStyle → LightGray, EdgeStyle->Directive[Black, Thick]]
TableForm[{tPred, tDepth, tDir, tDinast},
TableHeadings → {{"pred[i]", "depth[i]", "dir[i]", "dinast[i]"}, VertexList[gr]}]
edgestyle1 = {};
For[i = 1, i ≤ Length[arcsn], i++,
{x1, y1} = List@@arcsn[[i]];
For[j = 1, j ≤ Length[tree], j++,
{x2, y2} = List@@tree[[j]];
If[(x2 ∧ y1 = y2) ∨ (x1 = y2 ∧ x2 = y1),
AppendTo[edgestyle1, x1 → y1]]]]Print[Spanning tree- ]
```

```
Graph[arcsn, GraphLayout → CircularEmbedding,
VertexLabels → Placed[Name, Center], VertexSize → 0.3, VertexLabelStyle → 15,
EdgeShapeFunction → ( Arrow  ArrowSize → 0.06 ) , VertexStyle → LightGray,
EdgeStyle → {(#1 → Directive[Black, Thick]&)/@edgestyle1/. List → Sequence}]
Determining(k_, l_, pred_, dir_, depth_):=Module[{deltaC = {}},
i = k; j = l; While[i ≠ j, If[depth[[i]] > depth[[j]],
{
If[dir[[i]] = 1,
AppendTo [deltaC, x_{pred[[i]]→i} → 1] ;
AppendTo[Ucycle, pred[[i]] → i],
AppendTo [deltaC, x_{i→pred[[i]]} → −1] ;
AppendTo[Ucycle, i → pred[[i]]]]; i = pred[[i]];
},
If[depth[[j]] > depth[[i]],
{
If[dir[[j]] = 1,
AppendTo [deltaC, x_{pred[[j]]→j} → −1] ;
AppendTo[Ucycle, pred[[j]] → j],
AppendTo [deltaC, x_{j→pred[[j]]} → 1] ;
AppendTo[Ucycle, j → pred[[j]]]];
j = pred[[j]];
},
{
If[dir[[i]] = 1,
AppendTo [deltaC, x_{pred[[i]]→i} → 1] ;
AppendTo[Ucycle, pred[[i]] → i],
AppendTo[deltaC, x_{i→pred[[i]]} → −1];
AppendTo[Ucycle, i → pred[[i]]]; ];
If[dir[[j]] = 1,
AppendTo [deltaC, x_{pred[[j]]→j} → −1] ;
AppendTo[Ucycle, pred[[j]] → j],
AppendTo [deltaC, x_{j→pred[[j]]} → 1] ;
AppendTo[Ucycle, j → pred[[j]]];
];
i = pred[[i]];
j = pred[[j]]; }
]]; ];
Return[deltaC]; ];

ComputeCharacteristicVectors[lUn_, lUt_]:=Module[{cv = {}},
dN0 = (x_{#1} → 0&) /@lUn;
```

```
For[n = 1, n ≤ Length[lUn], n++,
arc = lUn[[n]];
dN = dN0; dN[[n]] = x_arc → 1;
dTc = Determining(arc[[1]], arc[[2]], tPred, tDir, tDepth);
listUt0 = Complement[lUt, Ucycle];
Ucycle = {};
dT0 = (x_#1 → 0&) /@listUt0;
dGeneral = Join[dN, Sort[Join[dTc, dT0]]];
AppendTo[cv, {dGeneral}]; ];
Return[cv]; ];
Ucycle = {};
listUn = Complement[arcsn, listUt];
arcUnCount = Length[listUn];
listUt = Sort[listUt];
delta = Flatten[ComputeCharacteristicVectorslistUn, listUt), 1];
TableForm[Table[delta[[i, j]][[2]], {i, 1, arcUnCount}, {j, 1, edgeCount}],
TableHeadings → {listUn, Table[delta[[1, i]][[1]], {i, 1, edgeCount}]},
TableAlignments → Center]
(x_#1&) /@listUn

M = Select[Range[Length[b]], b[[#1]]=!=0&]
Print["M = ", M];
b = b/.x->f;
Print["b = ", b];
balanceEqs =
((Total[If[Or[MemberQ[M, #[[1]]], MemberQ[M, #[[2]]]], f_#, x]
&/@EdgeList[g, #_]] − Total[If[Or[MemberQ[M, #[[1]]],
MemberQ[M, #[[2]]]], f_#, x_#]&/@EdgeList[g, _#]])) ==
b[[#]]&/@VertexList[gr];
TableForm[balanceEqs]
incL = (IncidenceList[gr, #]&/@M)//Flatten
incL1 = DeleteDuplicates[incL];
Print[" : ", incL1]
b̄ = Fold[If[MemberQ [M, #2_[[1]]] , ReplacePart[#1, #2_[[2]] → #_[[#2[[2]]]] + f_#2],
ReplacePart[#1, #2_[[1]] → #_[[#2[[1]]]] − f_#2]]&, b, incL];
newB = b̄;
For[i = 1, i<=Length[newB], i++,
For[j = 1, j<=Length[newB[[i]]], j++,
If[newB[[i]][[j]][[1]]==2,
a = newB[[i]][[j]][[2]];
```

```
newB[[i]][[j]][[1]] = 1;
 AppendTo [newB[[i]], -f_{a[[2]][[2]]a[[2]][[1]]}]
];
];
b̄ = newB;
newGraph = VertexDelete[g, M];
GraphPlot[newGraph, EdgeStyle → Directive[Black, Thick],
VertexStyle → Directive[EdgeForm[Thick], White], MultiedgeStyle → .05,
VertexLabels->Placed["Name", Center], .
VertexSize->0.25,
VertexLabelStyle → Directive[Black, 20], MultiedgeStyle → .07]
b̄//forma
ii_i^+[g_]:=Cases[IncidenceList[g, i], u_v_/;u == i :→ v]
M' = Complement[VertexList[g], M]
edg = EdgeList[g]
M^+ = Select[edg, MemberQ[M', #[[1]]]&&MemberQ[M, #[[2]]]&]
Print ["Length[M^+] = ", Length [M^+]]
Print["Length[M] = ", Length[M]]
Print ["M^+ " = , M^+]
b1 = Fold[Module[ bb = #1, i = #2_{[[1]]}, k = #2_{[[2]]}},
(ReplacePart[bb, ((({# → bb_{[[#]]} + (p_{i#}/p_{ik}) f_{ik}, i → bb_{[[i]]} - (p_{i#}/p_{ik}) f_{ik}})&)/@
ii_i^+[newGraph])//Flatten])]&, b̄, M^+]
TableForm[b1]
GraphPlot[Fold[HighlightGraph[#1, u_v_/;u == #2,
GraphHighlightStyle → "White"]&, newGraph, #_{[[1]]}&/@M^+],
VertexLabels->Placed["Name", Center], .
VertexLabelStyle → Directive[Black, 20], VertexSize->0.25,
EdgeStyle → Directive[Black, Thick],
VertexStyle → Directive[EdgeForm[Thick], White],
MultiedgeStyle → .05]
g1 = Fold [EdgeDelete[#1, u_v_/;u == #2]&, newGraph, #_{[[1]]}&/@M^+] ;
GraphPlot[g1, EdgeStyle → Directive[Black, Thick],
VertexStyle → Directive[EdgeForm[Thick], White], MultiedgeStyle → .05,
VertexLabels->Placed["Name", Center], .VertexSize->0.25]
g = g1;
b = b1;
b//forma
AppendTo[b, -Total[b]];
replaceX[expr_] := expr/.{Subscript[x, i_DirectedEdge j_] :>
With[{k = Select[edgesForAdditionalEquations, #[[1]] == i&]},
```

```
If[k =! = {},
Module[{repl},
repl = (Subscript[f, iDirectedEdgek[[1]][[2]]] * Subscript[p,
iDirectedEdgej])/Subscript[p, iDirectedEdgek[[1]][[2]]];
repl],
Subscript[x, iDirectedEdgej]]};
getStartReplacement[Subscript[x, i_DirectedEdgej_]] := With[
{k = Select[edgesForAdditionalEquations, #[[1]] == i&]},
If[k =! = {},
Module[{repl},
repl = (Subscript[f, iDirectedEdgek[[1]][[2]]] * Subscript[p,
iDirectedEdgej])/Subscript[p, iDirectedEdgek[[1]][[2]]];
repl],
Subscript[x, iDirectedEdgej](*если подходящего k нет*)]};
getSubstitution[edge_] :=
SelectFirst[solutions, MatchQ[First[#], Subscript[x, edge]]&];
replaceDynamicX[expr_] :=
expr/.{Subscript[x, i_DirectedEdgej_] :>
With[{k = Select[newAdditionalEdges, #[[1]] == i&]},
If[k =! = {},
Module[{repl},
sub = getSubstitution[iDirectedEdgek[[1]][[2]]];
repl = (sub[[2]] * Subscript[p, iDirectedEdgej])/Subscript[
p, iDirectedEdgek[[1]][[2]]];
AppendTo[solutions, Subscript[x, iDirectedEdgej] → repl];
DeleteDuplicates[solutions]; repl], Subscript[x, iDirectedEdgej]}]};balanceEqs//forma;
balanceEqs = replaceX[balanceEqs];
systemWithData = balanceEqs;
uniqueX = DeleteDuplicates@
Select[Level[balanceEqs, {0, Infinity}],
MatchQ[#, Subscript[x, _DirectedEdge_]]&];
replacements = Table[el → getStartReplacement[el], {el, uniqueX}];
replacements =
Select[replacements, !
MatchQ[#[[2]], Subscript[x, _DirectedEdge_]]&];
Print["Подстановка дополнительных уравнений"]
systemWithData//forma
(*основной цикл программы*)
While[AnyTrue[systemWithData, !FreeQ[#, x]&],
(*Ищем решение на уравнениях, где слева есть x*)
eqWithX =
```

```
Select[systemWithData, !
FreeQ[First[#], x]&]; (*уравнения с X в левой части*)
vars = DeleteDuplicates@
Cases[eqWithX, Subscript[x, _DirectedEdge], Infinity];
sol = Reduce[eqWithX, vars];
If[Head[sol] === Or,
sol = First[sol]];
xSolutions =
Select[sol, !FreeQ[First[#], x]&&FreeQ[Last[#], x]&];
xSolutions = List@@ToRules[xSolutions];
Print[xSolutions];
(*Добавляем новые ребра и решения*)
solutions = Join[solutions, xSolutions];
newEdges = (First[#]/.
Subscript[x, i_DirectedEdge j_] :> iDirectedEdge j)&/@
xSolutions;
newAdditionalEdges = Join[newAdditionalEdges, newEdges];
(*Подставляем решения в систему уравнений*)
systemWithData = systemWithData/.solutions;
systemWithData = replaceDynamicX[systemWithData]; ];
```

Листинг программы (генерация графа)

```
vertexCount = 15;
edgeCount = 20;
numX = 6;
SeedRandom[];
baseEdges = Table[{i, (i mod vertexCount) + 1}, {i, 1, vertexCount}];
additionalEdgesNeeded = edgeCount − vertexCount;
existingSet = Association[];
Do[existingSet({e[[1]], e[[2]]}) = True, {e, baseEdges}];
extraEdges = {};
While[Length[extraEdges] < additionalEdgesNeeded,
u = RandomInteger[{1, vertexCount}];
v = RandomInteger[{1, vertexCount}];
If[u = v, Continue[]];
If[¬KeyExistsQ[existingSet, {u, v}]∧
¬KeyExistsQ[existingSet, {v, u}] ∧ ¬MemberQ[extraEdges, {u, v}]∧
¬MemberQ[extraEdges, {v, u}],
AppendTo[extraEdges, {u, v}];
existingSet({u, v}) = True; ]; ];
allEdges = Join[baseEdges, extraEdges];
If[Length[DeleteDuplicates[allEdges]]=!=Length[allEdges],
Print[ !]];
If[AnyTrue[allEdges, MemberQ[allEdges, Reverse[#1]]&],
Print[  !]];
graph = Graph[allEdges, DirectedEdges → True];
If[¬StronglyConnectedGraphQ(graph),
Print[   !]]; )
(∗Полный набор рёбер*)
allEdges = Join[baseEdges, extraEdges];
(∗Проверка на уникальность*)
If[Length[DeleteDuplicates[allEdges]] =! = Length[allEdges],
Print["Повторяющиеся рёбра!"]];
(∗Проверка на антипараллельные рёбра*)
If[AnyTrue[allEdges, MemberQ[allEdges, Reverse[#]]&],
Print["Найдены антипараллельные рёбра!"]];
(∗Построение графа и проверка связности*)
graph = Graph[allEdges, DirectedEdges → True];
Graph[graph, EdgeStyle -> Directive[Black, Thick],
```

```
VertexLabels -> Placed["Name Center],
VertexSize -> 0.5, VertexStyle -> Directive[EdgeForm[Thick], White],
VertexLabelStyle -> Directive[Black, 24],
GraphLayout -> "CircularEmbedding
VertexShapeFunction -> {xx_ :> If[MemberQ[xVertices, xx], "Square "Circle"]}]
If[!StronglyConnectedGraphQ[graph], Print["Граф не сильно связный!"]];
xVertices = RandomSample[Range[vertexCount], numX];
header = {/*|I|*/ <> ToString[vertexCount],
/*|U|*/ <> ToString[Length[allEdges]]};
edgesStr = ({<> ToString[#1[[1]]] <>, <> ToString[#1[[2]]] <>}&)
/@allEdges;
bValues = Table[/*b_ <> ToString[i] <> */ <>
If[MemberQ[xVertices, i], x, 0], {i, 1, vertexCount}];
outputLines = Join[header, edgesStr, bValues];
notebookDir = NotebookDirectory[];
Export[FileNameJoin[{notebookDir, graph.txt}], outputLines, Text]
```