УДК 004.94:620.9

РАСЧЕТ ДОПУСТИМОСТИ ОПЕРАТИВНОГО ПЕРЕКЛЮЧЕНИЯ В СЕТЯХ ЭЛЕКТРОСНАБЖЕНИЯ.

А.Н. Михайлов, Иконников С.Е.

Аннотация

Расчет допустимости оперативного переключения в сетях электроснабжения является важной функцией диспетчерского управления. Предварительная оценка необходима для определения допустимости оперативного переключения, что обеспечивает уменьшение влияния человеческого фактора и, в конечном итоге, повышение надежности электроснабжения. Рассмотрен один из вариантов решения задачи определения пропускной способности сети. Создана объектная модель графа электрической сети и на её основе реализован и исследован алгоритм Push-Relabel.

Ключевые понятия: оперативное переключение, пропускная способность, максимальный поток, граф, Push-Relabel.

В службах регионального и городского уровней диспетчерского управления распределения электроэнергии часто приходится принимать решение об оперативном переключении в сети электроснабжения – передачи части нагрузки с одного участка сети на другой. Прежде чем подключать нового потребителя, следует просчитать пропускную способность данной конфигурации сети, что необходимо для принятия решения о допустимости оперативного переключения.

Если производить подключение потребителя или оперативное переключение без расчета пропускной способности электрической сети, то это может привести к аварийному выходу из строя ее элементов. Обычно для предотвращения аварии ставят защитные элементы. Но даже их срабатывание приводит к длительному отсутствию электричества у потребителей.

Пропускная способность электрической сети – это максимальное значение силы тока, который данная сеть может пропустить для потребителей электрической энергии. Рассматриваются электрические сети городского масштаба, которые состоят из линий электропередачи (ЛЭП), трансформаторных подстанций, распределительных и секционирующих пунктов. Пропускная способность участка сети ограничивается пропускной способностью элементов сети.

Введем определение фидера: фидер – это множество объектов сети, электрически соединенных в данный момент времени.

Представим задачу расчета пропускной способности как задачу нахождения максимального потока[1]. Для примера рассмотрим участок электрической сети на рис. 1.

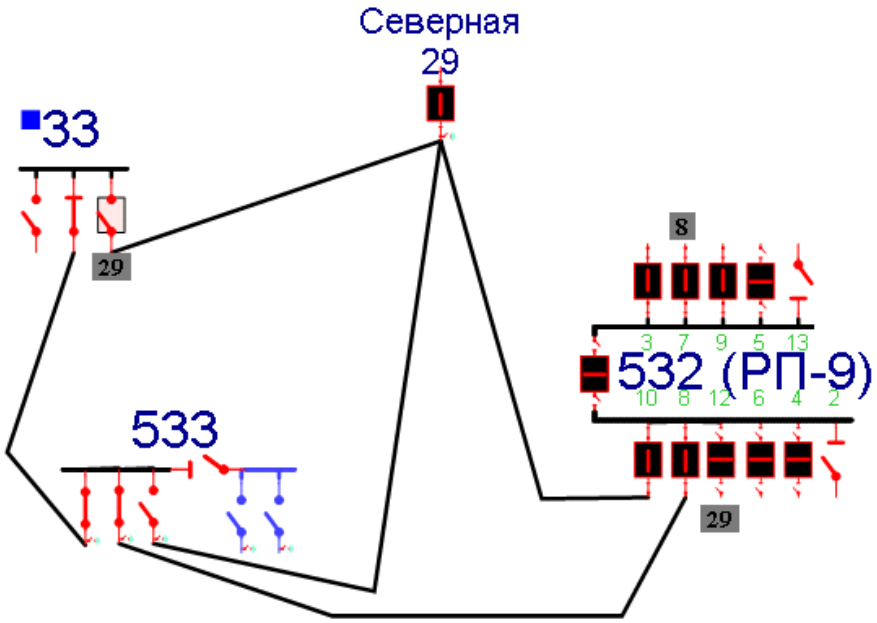

Рис. 1 – Пример участка электрической сети (фидера)

В данном примере изображены три трансформаторные подстанции (ТП) под номерами 33, 532, 533. Источником энергии для них является центра питания (ЦП) «Северная». ТП и ЦП соединены друг с другом посредством ЛЭП, изображенных на рисунке в виде прямых. В каждом из ТП имеется силовой трансформатор напряжения. К нему подключаются потребители электроэнергии.

Ячейка и ЛЭП являются составными объектами. Их пропускная способность будет равна минимальной пропускной способности составных элементов. Необходимо определить, каково максимальное значение тока, который может пропустить данная конфигурация сети.

Представленный фрагмент электрической сети и есть не что иное, как фидер. Рассмотрим его в виде графа (рис. 2). Конфигурация сети при переключениях постоянно трансформируется, поэтому фидер должен динамически изменяться.

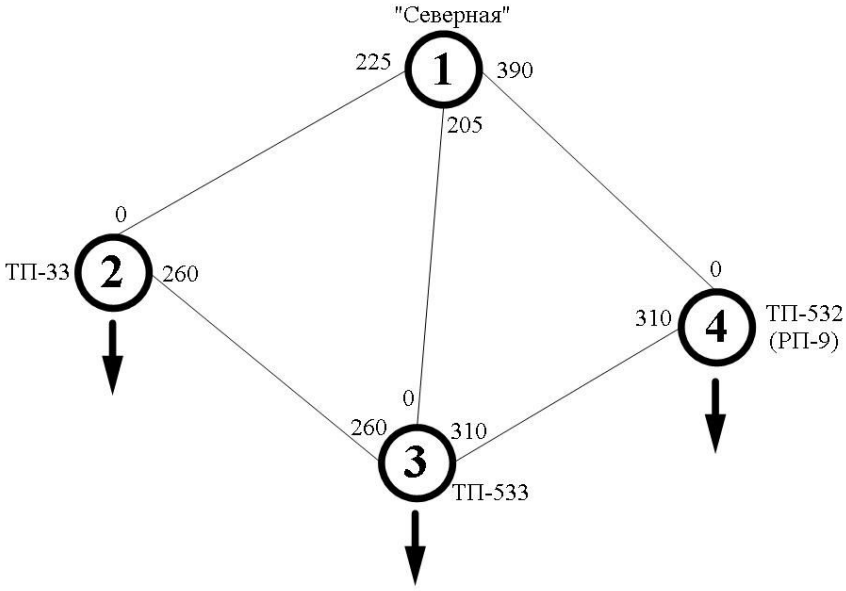

Рис. 2 – Представление электрической сети в виде графа

Вершинам графа сопоставлены ЦП и ТП, ребрам – совокупность объектов. В ребро могут быть включены в разных ситуациях ячейки, ЛЭП и трансформаторы. Например, в ребро 1–2 будут входить ЛЭП и ячейка, принадлежащая ТП-33. Каждому ребру назначена пропускная способность, равная минимальной из его элементов. Граф является направленным, и позиция обозначения пропускной способности определяет направление ребра. Значение пропускной способности в одном направлении может отличаться от значения в обратном.

Чтобы свести задачу расчета пропускной способности сети к задаче нахождения максимального потока, необходимо дополнить данный граф еще одной вершиной, которую назовем *суперстоком*, и добавить ребра от стоков в новой вершине (рис. 3). В данном примере понятие *суперстока* эквивалентно понятию потребителя.

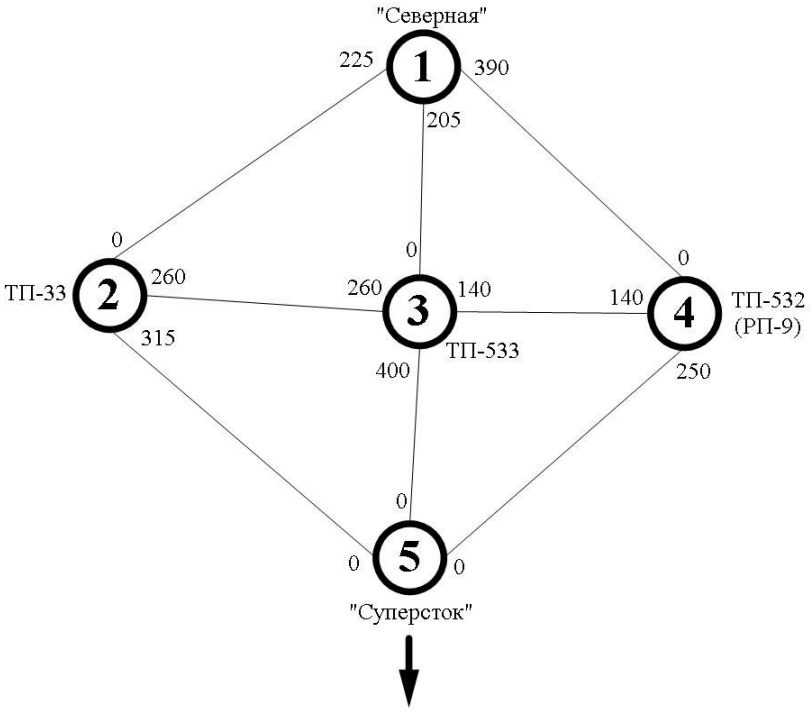

Рис. 3 – Модифицированный граф

Обычно ребрам, которые идут от стоков к *суперстоку*, присваивается пропускная способность, равная ∞ , чтобы они не влияли на результат, но в нашем случае правильней назначить им пропускную способность, равную мощности трансформаторов, так как именно они являются конечными элементами в цепочке от источника к *суперстоку*. После перечисленных преобразований имеем задачу нахождения максимального потока, которая определяет оптимальную пропускную способность (т.е. максимальный поток) между источником и стоком (в нашем случае *суперстоком*). Задача с нескольким источниками электрического тока сводится к данной путем добавления одного общего *суперисточника*.

Задача о максимальном потоке изучается сравнительно недавно. Интерес к ней обусловлен широким практическим применением – на коммуникационных, транспортных, электрических сетях, при моделировании различных процессов физики и химии, в некоторых операциях над матрицами,

для решения родственных задач теории графов и даже для поиска Web-групп в Internet. Первое решение основывалось на simplex-методах линейного программирования, что было крайне неэффективно. Форд Л. Р. и Фалкерсон Д. Р. предложили рассматривать для решения задачи о максимальном потоке ориентированную сеть и искать решение с помощью итерационного алгоритма. В течение 20 лет все передовые достижения в исследовании данной задачи базировались на их методе. В 1970 г. наш соотечественник Е.А. Диниц предложил решать задачу с использованием вспомогательных бесконтурных сетей и псевдомаксимальных потоков, что намного увеличило быстродействие разрабатываемых алгоритмов. В 1974 г. А.В. Карзанов улучшил метод Диница, введя такое понятие, как предпоток. Алгоритмы Диница и Карзанова, как и исследования Форда и Фалкерсона, внесли огромный вклад в решение данной проблемы. На основе их методов 15 лет достигались наилучшие оценки быстродействия алгоритмов. В 1986 г. появился третий метод, который также можно отнести к фундаментальным. Этот метод был разработан А.В. Голдбергом и Р.Е. Таряном и получил название Push-Relabel. Для нахождения максимального потока, он использует предпотоки и метки, изменяемые во время работы алгоритма. Push-Relabel алгоритмы очень эффективны, и исследуются до сих пор. И, наконец, в 1997 г. А.В. Голдберг и С. Рао предложили алгоритм, присваивающий дугам неединичную длину. Это самый современный из алгоритмов. Асимптотическая оценка его быстродействия превзошла O(nm), где n и m – количество вершин и ребер соответственно.

Разберем алгоритм Push-Relabel как наиболее простой с точки зрения реализации и подходящий по скорости. В качестве примера будем рассматривать граф на рис. 3. Введем некоторые определения. Для ребра (i, j), где i < j, используем запись (Cij, Cji) для представления пропускных способностей в направлениях i → j и j → i соответственно. Во избежание недоразумений на схеме сети Cij будем располагать на ребре (i, j) ближе к узлу i, а Cji – ближе к узлу j (рис. 4).

Рис. 4 – Обозначение пропускной способности ребра

Идея первого алгоритма состоит в нахождении сквозных путей с положительными потоками от источника к стоку.

Рассмотрим ребро (i, j) с (начальной) пропускной способностью (Cij,Cji). В процессе выполнения алгоритма части этих пропускных способностей «забираются» потоками, проходящими через ребро, в результате каждое из них будет иметь остаточную пропускную способность. Будем использовать запись (cij, cji) для представления остаточных пропускных способностей. Сеть, где все ребра имеют остаточную пропускную способность, назовем остаточной.

Для произвольного узла j, получающего поток от узла i, определим метку [aj, i], где aj – величина потока, протекающего от узла j к узлу i.

Алгоритм нахождения максимального потока предполагает выполнение следующих действий.

Шаг 1. Для всех ребер (i, j) положим остаточную пропускную способность равной первоначальной пропускной способности, т.е. приравняем (cij, cji) = (Cij, Cji). Назначим a1 = ∞ и пометим узел 1 меткой [∞ , –]. Полагаем i = 1 и переходим ко второму шагу.

Шаг 2. Определяем множество Si как множество узлов j, в которые можно перейти из узла i по ребру с положительной остаточной пропускной способностью (т.е. cij > 0 для всех j, принадлежащих Si). Если Si не пустое множество, выполняем третий шаг, в противном случае переходим к шагу 4.

Шаг 3. В множестве Si находим узел k, такой, что cik = max {cij} для всех j, принадлежащих Si. Положим ak = cik и пометим узел k меткой [ak, i]. Если последней меткой помечен узел стока (т.е. если k = n), сквозной путь найден, и мы переходим к пятому шагу.

Шаг 4 (Откат назад). Если i = 1, сквозной путь невозможен, и мы переходим к шагу 6. Если i не равно 1, находим помеченный узел r, непосредственно предшествующий узлу i, и удаляем узел i из множества узлов, смежных с узлом r. Полагаем i = r и возвращаемся ко второму шагу.

Шаг 5 (Определение остаточной сети). Обозначим через Np = {1, k1, k2, …, n} множество узлов, через которые проходит p-й найденный сквозной путь от узла источника (узел 1) до узла стока (узел n). Тогда максимальный поток, проходящий по этому пути, вычисляется как fp = min {a1, ak1, ak2, ..., an}.

Остальные пропускные способности ребер, составляющих сквозной путь, уменьшаются на величину fp в направлении движения потока и увеличиваются на эту же величину в противоположном направлении. Таким образом, для ребра (i, j), входящего в сквозной путь, текущие остаточные стоимости (cij, cji) изменятся следующим образом:

- (cij – fp, cji + fp), если поток идет от узла i к узлу j,
- (cij + fp, cji – fp), если поток идет от узла j к узлу i.

Далее восстанавливаем все узлы, удаленные на шаге 4, полагаем i = 1 и возвращаемся ко второму шагу для поиска нового сквозного пути.

Шаг 6 (Решение).

- При m найденных сквозных путях максимальный поток вычисляется по формуле F = f1 + f2 + ... + fm.
- Имея значения начальных (Cij, Cji) и конечных (cij, cji) пропускных способностей ребра (i, j), можно вычислить оптимальный поток через это ребро следующим образом. Положим (a,b) = (Cij – cij, Cji – cji). Если a > 0, поток, проходящий через ребро (i, j), равен a. Если же b > 0, тогда поток равен b. (Случай, когда одновременно a > 0 и b > 0, невозможен.)

Приведем пример решения.

Итерация 1. Графическая иллюстрация к первой итерации приведена на рис. 4.

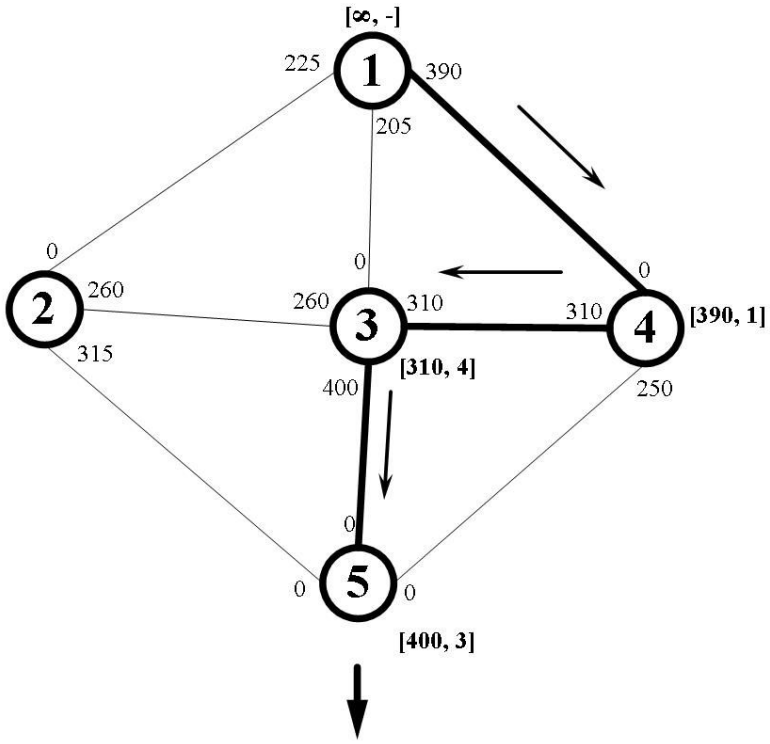

Рис. 4 – Итерация 1

Положим остаточные пропускные способности (cij, cji) всех ребер равными первоначальным пропускным способностям (Cij, Cji).

Шаг 1. Назначаем a1 = ∞ и помечаем узел 1 меткой [∞ , –]. Полагаем i = 1.

Шаг 2. S1 = [2, 3, 4] (множество не пустое).

Шаг 3. k = 4, поскольку c14 = max {c12, c13, c14} = max {225, 205, 390} = 390. Назначаем a4 = c14 = 390 и помечаем узел 4 меткой [390, 1]. Полагаем i = 4 и возвращаемся к шагу 2.

Шаг 2. S2 = [3, 5].

Шаг 3. k = 3 и a3 = c45 = max {310, 250} = 310. Помечаем узел 3 меткой [310,4].

Шаг 2. S3 = [2, 5]. Шаг 4 не берем, поскольку вершина уже помечена.

Шаг 3. k = 5 и a5 = c35 = max{260, 400}. Помечаем узел 5 меткой [400, 1]. Получен сквозной путь. Переходим к шагу 5.

Шаг 5. Сквозной путь определяем по меткам, начиная с узла 5 и заканчивая узлом 1: (5) → [400, 3] → (3) → [310, 4] → (4) → [390, 1] → (1). Таким образом, N1 = {1, 4, 3, 5} и f1 = min {a1, a4, a3, a5} = {∞ , 390, 310, 400} = 310. Вычисляем остаточные пропускные способности вдоль пути N1:

(c14, c41) = (390 – 310, 0 + 310) = (80, 310),

(c43, c34) = (310 – 310, 310 + 310) = (0, 620),

(c35, c53) = (400 – 310, 0 + 310) = (90, 310).

Итерация 2. Графическая иллюстрация ко второй итерации приведена на рис. 5.

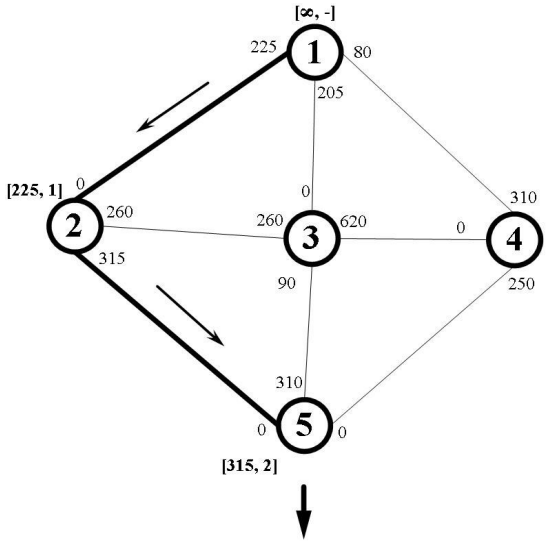

Рис. 5 – Итерация 2

Шаг 1. Назначаем a1 = ∞ и помечаем узел 1 меткой [∞ , –]. Полагаем i = 1.

Шаг 2. S1 = [2, 3, 4].

Шаг 3. k = 2, поскольку c12 = max {c12, c13, c14} = 225. Назначаем a2 = c12 = 225 и помечаем узел 2 меткой [225, 1]. Полагаем i = 2 и возвращаемся к шагу 2.

Шаг 2. S2 = [3, 5].

Шаг 3. k = 5 и a5 = c25 = max {315, 260} = 315. Помечаем узел 5 меткой [315, 2]. Получен сквозной путь. Переходим к шагу 5.

Шаг 5. Сквозной путь определяем по меткам, начиная с узла 5 и заканчивая узлом 1: (5) → [315, 3] → (3) → [225, 2] → (1). N2 = {1, 2, 5} и f2 = min {a1, a2, a5} = {∞ , 225, 315} = 225. Вычисляем остаточные пропускные способности вдоль пути N2:

(c12, c21) = (225 – 225, 0 + 225) = (0, 225),

(c25, c52) = (315 – 225, 0 + 225) = (90, 225).

Итерация 3. Графическая иллюстрация к третьей итерации приведена на рис. 6.

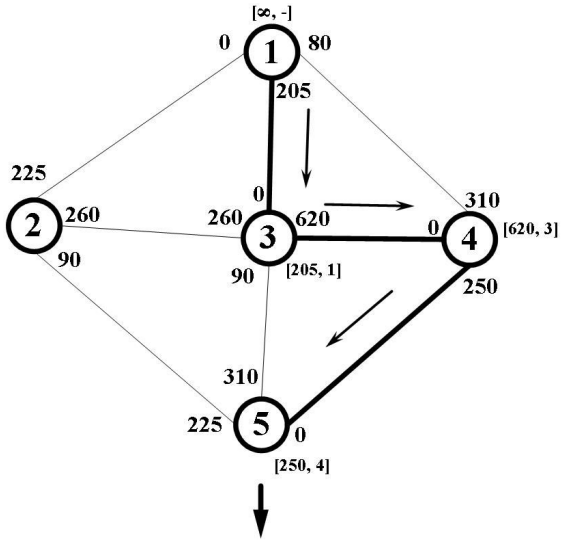

Рис. 6 – Итерация 3

Шаг 1. Назначаем a1 = ∞ и помечаем узел 1 меткой [∞ , –]. Полагаем i = 1.

Шаг 2. S1 = [2, 3, 4].

Шаг 3. k = 3, поскольку c13 = max {c12, c13, c14} = 205. Назначаем a3 = c13 = 205 и помечаем узел 3 меткой [205, 1]. Полагаем i = 3 и возвращаемся к шагу 2.

Шаг 2. S2 = [2, 4, 5].

Шаг 3. k = 4 и a4 = c34 = max {620, 260, 90} = 620. Помечаем узел 5 меткой [620, 3].

Шаг 2. S3 = [5]. Вершину 3 не берем в расчет, поскольку путь к ней невозможен. Остаточная пропускная способность равна 0.

Шаг 3. k = 5 и a5 = c45 = max{250} = 250. Помечаем узел 5 меткой [250, 4]. Получен сквозной путь. Переходим к шагу 5.

Шаг 5. Сквозной путь определяем по меткам, начиная с узла 5 и заканчивая узлом 1: (5) → [250, 4] → (4) → [620, 3] → (3) → [205, 2] → (1). N3 = {1, 2, 5} и f3 = min {a1, a3, a4, a5} = {∞ , 205, 620, 250} = 205. Вычисляем остаточные пропускные способности вдоль пути N2:

(c13, c31) = (205 – 205, 0 + 205) = (0, 205),

(c34, c43) = (620 – 225, 0 + 205) = (0, 205),

(c45, c54) = (250 – 205, 0 + 205) = (45, 205).

Итерации 4 и 5 изображены на рис. 7.

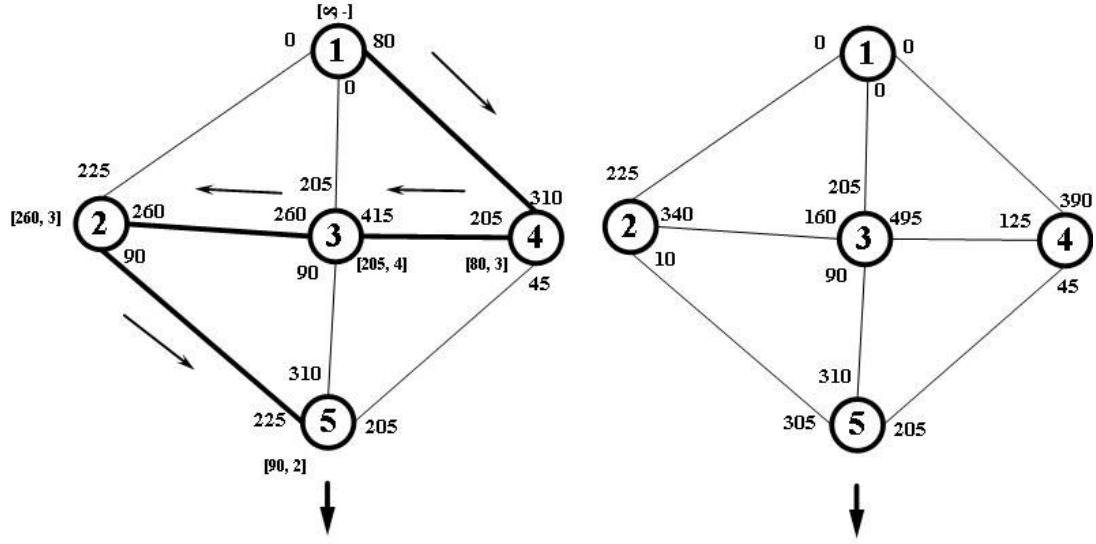

Рис. 7 – Итерации 4 и 5

На четвертой итерации получен путь N4 = {1, 4, 3, 2, 5} с f4 = 80. На пятой итерации новые сквозные пути невозможны, поскольку все ребра, исходящие из узла 1, имеют нулевые остаточные пропускные способности. Переходим к шагу 6 для определения решения.

Шаг 6. Максимальный объем потока в сети равен: F = f1 + f2 + f3 +f4 = 310 + 225 + 205 + 80 = 820 Ампер. Значения потоков по различным ребрам вычисляются путем вычитания последних значений остаточных пропускных способностей из первоначальных значений пропускных способностей (Cij, Cji).

Приведенный пример достаточно прост, чтобы наглядно продемонстрировать правильность алгоритма. Рассмотрим более сложный

пример (рис. 8), показывающий, что пропускная способность графа сети не всегда равна срезу ребер от источника.

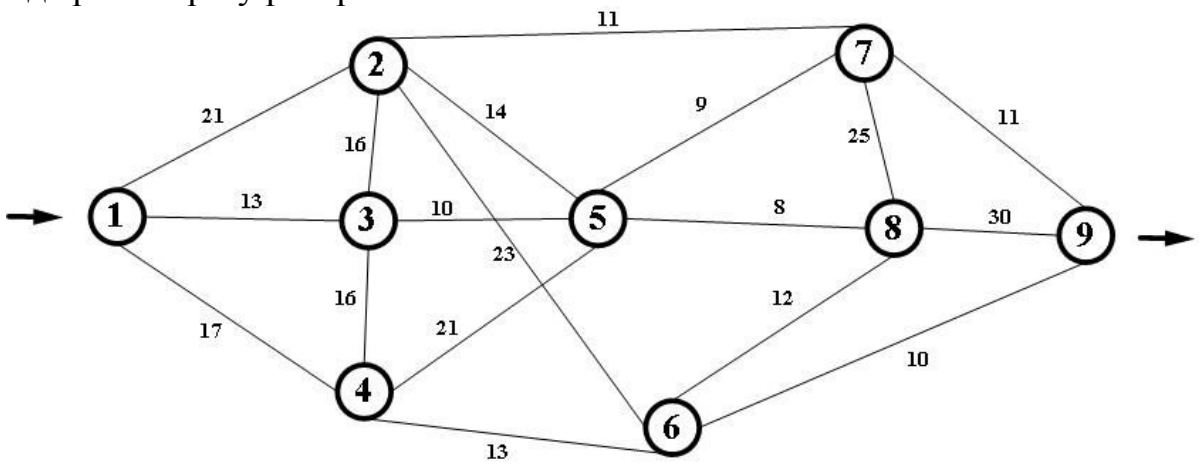

Рис. 8 – Пример участка сети, пропускная способность которого меньше среза при источнике.

Сумма пропускных способностей ребер, исходящих из вершины 1, которая является источником, равна 51. Сумма пропускных способностей ребер, входящих в вершину 9, которая является истоком, также равна 51. Но в данном примере общая пропускная способность всей сети меньше этого значения и равна 50. Если же диспетчер, ориентируясь на срезы при источнике и истоке, переведет на данный участок нагрузку большую, нежели она сможет пропустить, например 51, то сработают защитные установки и потребитель останется без электроэнергии. А если защитное оборудование не установлено, то возможен выход электроустановок из строя.

В данный фидер входит 9 трансформаторных подстанций, и ручной подсчет пропускной способности уже не представляется возможным, но в реальной ситуации их число может быть гораздо больше.

Программная реализация осуществлена с использованием универсальной объектной модели графа электрической сети[3]. Введем новый класс TMaxFlowProcessor в объектную модель. У данного класса будет единственный видимый метод:

function MaxFlow(aGraph: TGraph; aSource, aDestination: TVertex): Double;

Он возвращает значение максимального потока. Опишем алгоритм в виде диаграммы деятельности UML (рис. 9). Из диаграммы видно, что алгоритм ищет сквозные пути от источника к стоку. Каждый найденный сквозной путь обладает пропускной способностью. Найденный поток «вычитается» из графа в прямом направлении и «прибавляется» в обратном. Тем самым получается остаточный граф. Далее поиск осуществляется уже по остаточному графу до тех пор, пока сквозных потоков уже не останется. Особо интересен момент, когда алгоритм доходит до тупиковой вершины, из которой перейти дальше нельзя. Приходится возвращаться назад. Таким образом, алгоритм может вернуться в самый исток. Именно тогда он и прекратит свою работу.

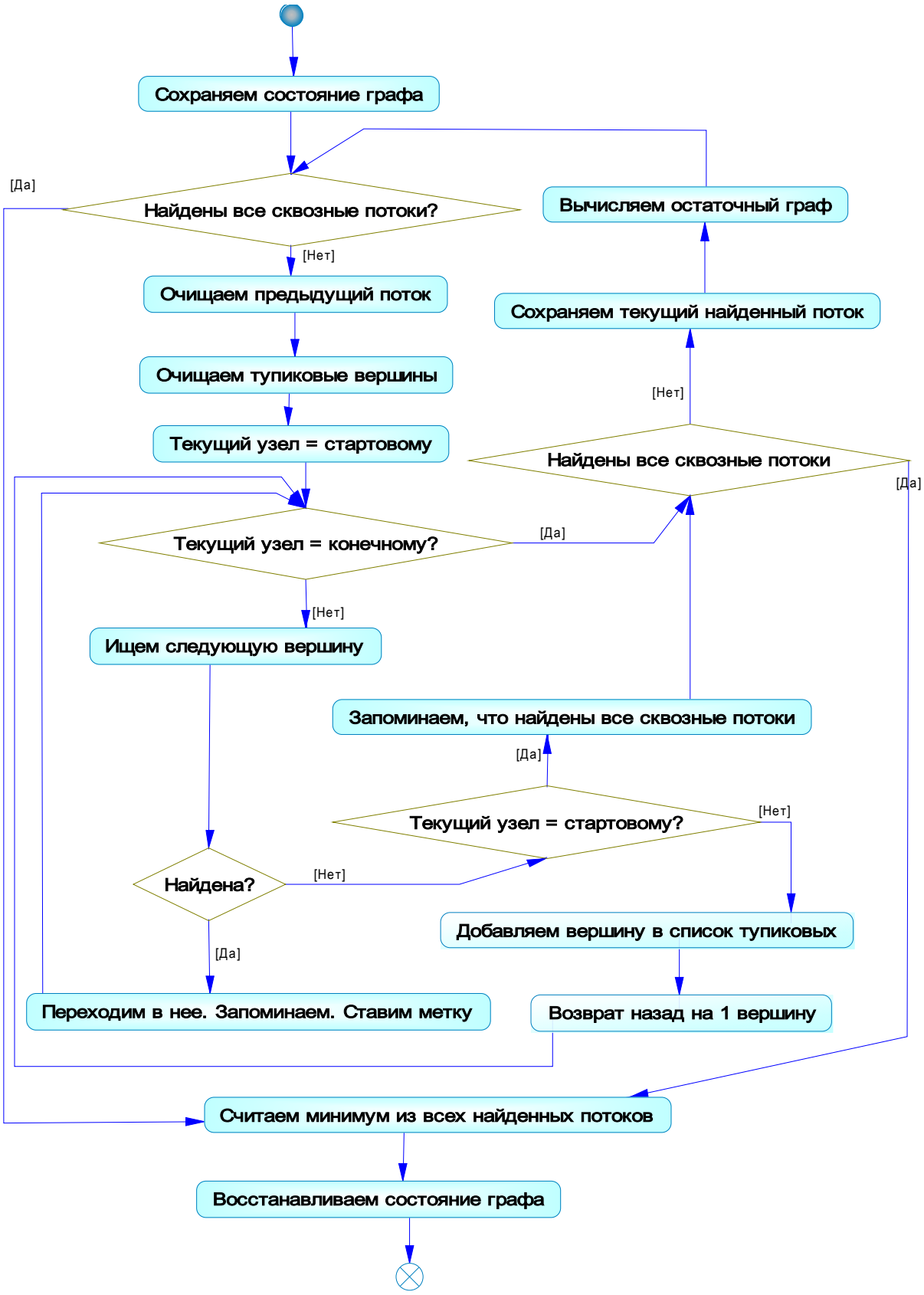

Рис. 9 – Диаграмма UML алгоритма нахождения максимального потока

Построим график зависимости скорости алгоритма от количества вершин и дуг (рис. 10).

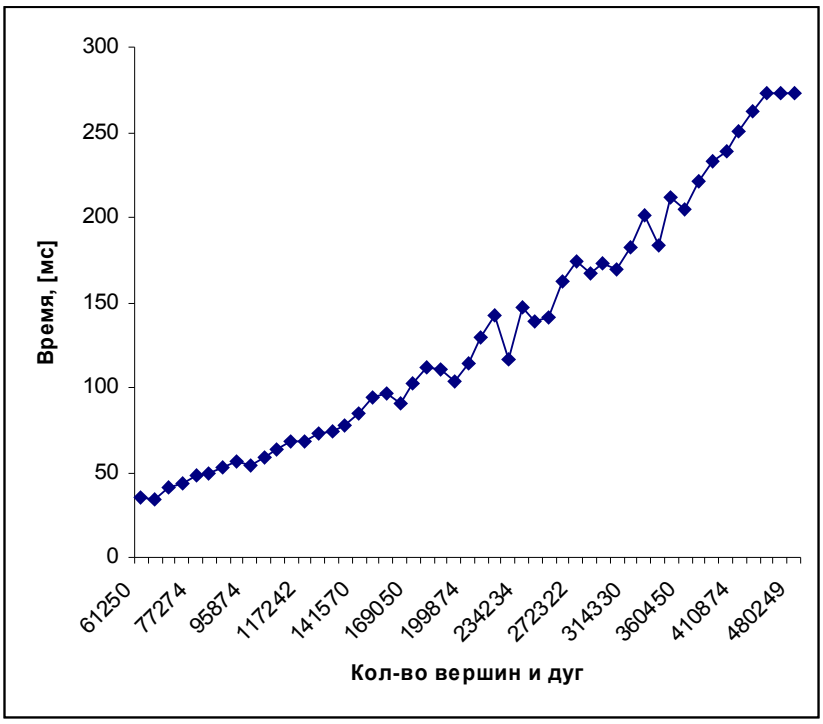

Рис. 10 – Скорость работы алгоритма

Данное испытание проводилось на плотном графе, т.е. каждая вершина была связана со всеми остальными в графе. Веса дугам присваивались случайным образом в диапазоне от 1 до 1000. Количество вершин и дуг постепенно увеличивалось в цикле. По оси абсцисс откладывается произведение количества дуг и количества вершин. Для каждого из значений по оси абсцисс испытание проводилось сто раз, и на графике отображено среднее значение времени, которое было затрачено алгоритмом на вычисление максимального потока. Как видно из графика, скорость возрастает линейно, что соответствует асимптотической оценке производительности O(nm). По оси ординат обозначено время в миллисекундах. Результаты записывались в базу данных, откуда импортировались в Microsoft Excel, в котором строился данный график.

Ниже приводится реализация алгоритма без подпрограмм:

```
function TMaxFlowProcessor.MaxFlow(aGraph: TGraph; aSource,
  aDestination: TVertex): Double;
begin
  Result := 0;
  SaveGraphState(aGraph);
  while not FIsNoAnyFlow do begin
    ClearFlowAndSi;
    while FCurrUzel.Vertex <> FStok do begin
      FNextVertex := GetNextVertex;
      if Assigned(FNextVertex) then begin
        AddNewVertexToFlow;
        MoveToNextVertex;
```

```
      end else begin
        if FCurrUzel.Vertex = FIstok then begin
          FIsNoAnyFlow := True;
          Break;
        end else
          ReturnBack;
      end;
    end;
    if FIsNoAnyFlow then
      Result := CalculateMaxFlow
    else
      CalculateRemainGraph(FFlow, FFlowResultI.C);
  end;
  RestoreGraphState();
end;
```

Выводы

1. Алгоритм Push-Relabel для расчета пропускной способности электрической сети обеспечивает быстродействие, необходимое для его применения в автоматизированных системах управления городской диспетчерской службы распределения электроэнергии.
2. Реализация алгоритма в виде шаблона объектной модели обеспечивает ее расширяемость и масштабируемость.

ПРИМЕЧАНИЯ

[1]Алгоритмы нахождения максимального потока. – URL: <http://algolist.ru/maths/graphs/maxflows/>

[2]Михайлов А. Н. Разработка универсальной объектной модели графа посредством тестирования / А Н. Михайлов, С. А. Ф. Федосин // Материалы XIII Научной конференции молодых ученых, аспирантов и студентов Морд. гос.ун-та им. Н. П. Огарева. В 2 ч. – Саранск, 2008. – Ч.2: Естеств. науки. – С. 289 – 296.