МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение
высшего образования «Национальный исследовательский Нижегородский
государственный университет им. Н.И. Лобачевского»

Институт информационных технологий, математики и механики

Н.В. Старостин

МОДЕЛИ И МЕТОДЫ УПОРЯДОЧЕНИЯ РАЗРЕЖЕННЫХ МАТРИЦ

Учебно-методическое пособие

Рекомендовано методической комиссией ИИТММ
для студентов ННГУ, обучающихся по направлениям подготовки
09.03.03 «Прикладная информатика» и
09.04.03 «Прикладная информатика»

Нижний Новгород
2018

УДК 519.85 (075.8)
ББК В185.7
С 77

Н.В. Старостин МОДЕЛИ И МЕТОДЫ УПОРЯДОЧЕНИЯ РАЗРЕЖЕННЫХ МАТРИЦ. Учебно-методическое пособие. – Нижний Новгород: Нижегородский госуниверситет, 2018. – 16 с.

Рецензент: д.т.н., профессор В.Е. Турлапов

В данном учебно-методическом пособии излагаются основные понятия и методы решения задач упорядочения разреженных матриц.

Пособие предназначено для бакалавров и магистров направления подготовки «Прикладная информатика».

УДК 519.85 (075.8)
ББК В185.7
С 77

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Многие инженерные задачи связаны с решением систем линейных алгебраических уравнений (СЛАУ) вида $Ax = b$, где A – заданная разреженная положительно определенная симметрическая квадратная матрица порядка n, b – заданный действительный вектор размерности n, и x – искомый вектор размерности n. В основе эффективного метода (как прямого, так и итерационного) решения больших разреженных систем линейных уравнений лежит хранение и использование разреженных матриц. Центральной проблемой при прямом решении разреженных симметричных СЛАУ является заполнение. Эффективность решения задачи прямым методом существенно зависит от того, насколько удается уменьшить это заполнение — количество новых ненулевых элементов, появляющихся в матрице коэффициентов в результате факторизации. Таким образом, возникает три задачи: 1) выбор надлежащего упорядочения; 2) формирование подходящей схемы хранения; 3) реальные вычисления [1].

Существуют различные схемы хранения разреженных матриц, отличающиеся способом использования нулей. В некоторых случаях допускается хранение части нулей в обмен на упрощение схемы хранения; в других — используются все нули системы; в третьих — нули вообще не используются. Выбор метода хранения, существенно, влияет на запросы к памяти и на использование стратегии упорядочения. Кроме того, он оказывает существенное воздействие на реализацию разложения и решения, а, следовательно, на сложность программ и время исполнения. Наиболее распространенными являются ленточная и профильная схемы хранения разреженных матриц, а также разреженный строчный формат (CSR). В процессе построения множителей Холецкого [2,3] разреженная матрица обычно претерпевает заполнение. Для того чтобы появлялось как можно меньше новых ненулевых элементов, предложены различные способы нумерации переменных и переупорядочения уравнений, которые равносильны перестановке строк и столбцов исходной матрицы. К настоящему времени известно достаточно много об источниках возникновения заполнения, и для его уменьшения разработаны специальные процедуры. Все они основаны на использовании свободы в выборе главных элементов, которые в случае положительно определенной матрицы A могут выбираться на диагонали в любом порядке.

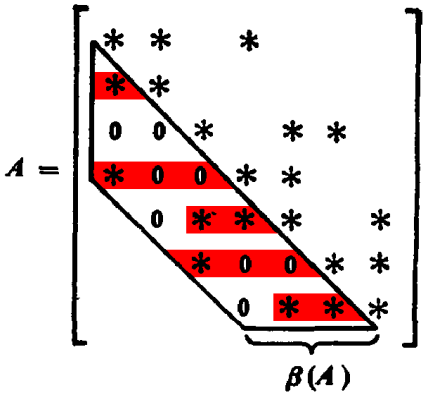
В основе ленточных и профильных методов лежит тот факт, что гауссово исключение может порождать новые ненулевые элементы лишь внутри ленты или профиля. Для уменьшения ширины ленты существует ряд быстрых алгоритмов. Наиболее распространенным и широко используемым, благодаря своей простоте является алгоритм Катхилла-Макки . Упорядочение, получаемое алгоритмом Катхилла-Макки, часто сильнее уменьшает профиль, чем первоначальное упорядочение, хотя ширина ленты остается неизменной [2,3]. Другими хорошо известными алгоритмами уменьшения профиля являются алгоритм Кинга [4] и алгоритм Слоуна [5].

Конструктивный детерминированный принцип построения упорядочения, который лежит в основе перечисленных выше алгоритмов, не позволяет их использовать повторно для получения лучших решений или поиска локальных оптимумов. С другой стороны итерационные улучшающие алгоритмы для задачи упорядочения оказываются слишком затратные с вычислительной точки зрения и не применимы к большеразмерным задачам.

В рамках данного учебного пособия рассматривается многоуровневый итерационный алгоритм, который комбинируя технику локального улучшения ширины ленты и редукции задачи, пытается получить лучшее упорядочение с точки зрения минимизации ширины ленты и профиля. Многоуровневость алгоритма позволят контролировать время его работы и затраты по памяти.

1 ЗАДАЧИ ПЕРУПОРЯДОЧЕНИЯ

Пусть $f_i(A) = \min\{ j \mid a_{ij} \neq 0 \}$, тогда $\beta_i(A) = i - f_i(A)$ называют i**-й шириной ленты**[1] (локальной шириной), $i = \overline{1,n}$, а $\beta(A) = \max\{ i - f_i(A) \mid i = \overline{1,n} \}$ - называют **шириной ленты**. Величину $\mathrm{Band}(A)$ определяют как область матрицы, удаленную от главной диагонали не более чем на $\beta(A)$ позиций (см. рис. 1).



i	$f_i(A)$	$\beta_i(A)$
1	1	0
2	1	1
3	3	0
4	1	3
5	3	2
6	3	3
7	5	2

Рис. 1. Лента матрицы

Профилем матрицы называют сумму всех локальных ширин $\mathrm{e}(A) = \sum_{i=1}^{n} \beta_i(A)$. Совокупность позиций от локальной ширины до главной диагонали называют **оболочкой** $\mathrm{Env}(A) = \{\{i, j\} \mid f_i \le j < i \}$. Справедлива формула: $|\mathrm{Env}(A)| = \sum_{i=1}^{n} \beta_i(A)$. На рис. 1 оболочка выделена красным цветом, профиль матрицы равен 11.

Упорядочивание матрицы – симметричное переупорядочивание строк и столбцов матрицы смежности. Очевидно, что упорядочиванием можно изменить значения ширины ленты и профиля. Сформулируем теперь рассматриваемую задачу явно. Пусть требуется решить систему $Ax = b$ с разреженной симметрической матрицей A. Если P — матрица перестановки, так что $PP^T = I$, то систему линейных уравнений можно переписать в виде $PAP^T Px = Pb$ или, если обозначить $y = Px$ и $c = Pb$, в виде $By = c$, где $B = PAP^T$ – переупорядоченная форма матрицы A, также являющаяся разреженной и симметрической. Кроме того, если матрица A положительно определена, то B также будет положительно определенной. С другой стороны, степень заполнения и число операций при факторизации матрицы B оказываются существенно зависящими от

[1] Здесь правильнее считать ее полушириной, но мы в силу симметричности матриц будем использовать именно это определение

перестановки P. Наша задача заключается в отыскании матрицы P (упорядочения строк и столбцов матрицы A).

$$\beta(PAP^T) \underset{P}{\rightarrow} \min \qquad (1)$$

$$\mathrm{Env}(PAP^T) \underset{P}{\rightarrow} \min \qquad (2)$$

Задачу (1) будем называть задачей минимизации ширины ленты матрицы (ЛЕНТА). Задачу (2) будем называть задачей минимизации профиля матрицы (ПРОФИЛЬ).

Для матрицы A порядка n существует $n!$ различных упорядочений. Указанные задачи является NP-трудными [7]. Существующие процедуры являются эвристическими и не гарантирующих достижения точного минимума этих величин.

На начальном этапе разряженная матрица A неупорядочена и может занимать много места. На этапе упорядочения нам нет нужды знать о конкретных значениях матрицы. Достаточно знать факт, есть ли в заданной строке и заданном столбце ненулевой элемент. В этом случае матрице ассоциируется помеченный граф $G = G(A) = (V, E)$. Для такого графа понятия ленты и профиля совпадают с определениями ленты и профиля матриц. Например, ширина ленты помеченного графа G можно определить так $b(G) = \max\{|i-j| \mid \{v_i, v_j\} \in E\}$. При выборе системы хранения графа в виде списков смежности нулевые элементы матрицы в принципе не хранятся. Однако, для такой формы представления матриц матричные алгоритмы непригодны. В силу этого, для упорядочения используются алгоритмы на графах, а теория графов представляет собой чрезвычайно полезный инструмент для анализа алгоритмов обработки разреженных матриц. Сформулируем постановки задач распознавания и соответствующие им экстремальные задачи на графах ([7]).

ЛЕНТА (Задача ТГ 40 – «ширина» в [7]). Задача распознавания. Для $(n,m)-$графа $G = (V, E)$ и натурального $B \le n$ спрашивается: существует ли линейное упорядочение множества V, такое, что его ширина не более B. Иными словами: существует ли такая взаимно-однозначная функция $f : V \leftrightarrow \{1,...,n\}$, такая, что для всех $\{u,v\} \in E$ выполнено $|f(u) - f(v)| \le B$? Оптимизационная формулировка задачи выглядит так: Требуется найти такое упорядочение f множества вершин графа, обеспечивающего минимальную ширину ленты.

Экстремальная задача. Требуется найти такое упорядочение f множества вершин графа ($f = (f_1,..., f_n)$ – перестановка из номеров вершин $1,...,n$), обеспечивающего минимальное значение ширины ленты:

$$F(f) = \max_{\{u,v\} \in E} |f(u) - f(v)| \to \min . \quad (3)$$

Задача (3) остается сложной [7] даже для деревьев со степенью вершин не более 3. Эта задача является сложной и для случая орграфа. Задача (3) соответствует минимизации ширины ленты матрицы смежности графа путем одновременных перестановок столбцов и строк.

ПРОФИЛЬ (Задача ТГ 42 – «оптимальное линейное упорядочение» в [7]). Задача распознавания. Для (n,m) – графа $G=(V,E)$ и натурального $B \le n$ спрашивается: существует ли линейное упорядочение множества V, такое, что его профиль не более B. Иными словами: существует ли такая взаимно-однозначная функция $f: V \leftrightarrow \{1,...,n\}$, такая, что $\sum_{\{u,v\}\in E} |f(u)-f(v)| \le B$?

Экстремальная задача. Требуется найти такое упорядочение f множества вершин графа, обеспечивающего минимальное значение профиля:

$$F(f)=\sum_{\{u,v\}\in E} |f(u)-f(v)| \to \min . \quad (4)$$

Задача (4) остается сложной [7] даже для двудольных графов и разрешима за полиномиальное время для деревьев. Задача (4) соответствует минимизации профиля матрицы смежности графа путем одновременных перестановок столбцов и строк.

2 АЛГОРИТМЫ МИНИМИЗАЦИИ ЛЕНТЫ И ПРОФИЛЯ

Для того, чтобы описать суть работы известных конструктивных алгоритмов введем ряд определений. **Окружением** $N(v)$ вершины называют множество (кроме самой вершины v) всех смежных вершин с v. **Окружением множества** вершин $N(M)$ $M \subseteq V$ называют множество всех смежных вершин с вершинами из M, кроме самих вершин множества M: $N(M) = \left(\bigcup_{v \in M} N(v) \right) \setminus M$.

Множество $N(\{v_1,...,v_i\})$ называют i-м **фронтом помеченного графа**, а число $w_i(G) = |N(\{v_1,...,v_i\})|$ – i-й **шириной фронта**. Для матрицы A ширина фронта $w_i(A) = |\{k \mid k > i \ \& \exists\, j \le i : a_{kj} \neq 0\}|$ – есть число строк оболочки A, которые пересекают i-й столбец (см. рис. 2).

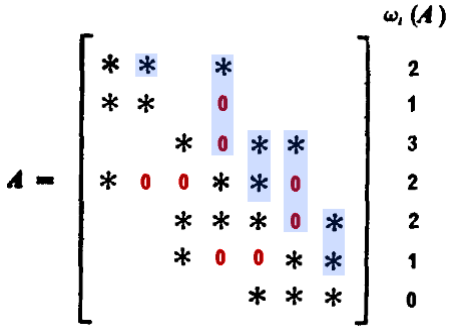


Рис.2. Локальные фронты матрицы (графа)

Для графа G, с матрицей смежности $A = (a_{ij})$, если $i < j$, то $\{j,i\} \in \mathrm{Env}(A) \Leftrightarrow v_j \in N(\{v_1,...,v_i\})$. Отсюда следует, что $w_i(A) = w_i(G)$ и $|\mathrm{Env}(A)| = \sum_{i=1}^{n} \omega_i(A)$.

Равенство значений профиля и фронта (сумма локальных фронтов) активно используется в конструктивных методах упорядочения. Пусть конструктивный алгоритм сделал $i-1$ шаг и пронумеровал $i-1$ вершину. На i-м шаге алгоритм выбирает очередную вершину, руководствуясь простым правилом. Выбирается та вершина, которая минимально увеличивает частично построенный профиль. Это можно делать анализируя значения i-й ширины ленты b_i, либо значения i-й ширины фронта w_i. Рассмотрим конкретные алгоритмы.

2.1 УПОРЯДОЧЕНИЕ КАТХИЛЛА-МАККИ

Суть алгоритма Катхилла-Макки (CM) [3,4] в упорядочении согласно уровням смежности. Алгоритм начинает со стартовой вершины (корень структуры уровней смежности). В качестве стартовой выбирается псевдопериферийная вершина. Стартовая вершина включается в первый уровень смежности и нумеруется (приписывается номер 1).

Для построения очередного уровня смежности алгоритм последовательно в порядке нумерации перебирает вершины последнего полностью построенного уровня смежности. Для каждой вершины формирует список непронумерованных и смежных ей вершин. Далее список упорядочивается по возрастанию локальных степеней вершин. В очередной уровень смежности вершины из списка включаются (и нумеруются) согласно возрастанию локальных степеней вершин.

Алгоритм последовательно формирует уровни смежности (нумерует вершины) до тех пор, пока не переберет все вершины компоненты связности. Если граф не является связным, то описанная схема применяется отдельно к каждой из его связных компонент.

Алгоритм CM, упорядочивая вершины по возрастанию локальных степеней, фактически для текущей строки матрицы уплотняет ненулевые значения во фронтальной части (уменьшает ширину фронта). Это один из самых быстрых и нетребовательных к памяти алгоритмов.

В приложении 1 приведен программный код на языке C# построения структуры смежности, поиска псевдопериферийной вершины и алгоритма CM.

2.2 ОБРАТНОЕ УПОРЯДОЧЕНИЕ КАТХИЛЛА-МАККИ

Показано, что если упорядочение CM обратить (перенумеровать в обратном порядке), то ширина ленты остается неизменной, однако профиль может уменьшиться (не увеличивается) [2,3]. На этом свойстве упорядочения CM основан обратный алгоритм Катхилла-Макки (RCM) [2,3], который находит упорядочение CM, а затем его обращает.

В приложении 1 приведен программный код на языке C# алгоритма RCM.

2.3 УПОРЯДОЧЕНИЕ КИНГА

Алгоритм Кинга (KING) [4] начинает нумерацию с вершины, имеющей минимальную степень. Алгоритм на каждом шаге разбиваем все множество вершин на три подмножества: A, B и C. Множество A состоит из всех уже пронумерованных вершин графа. Множество B определяется как смежное по отношению к A, т. е. $B = N(A)$ и, таким образом, состоит из всех вершин, которые смежны какой-либо вершине из A. Множество C состоит из всех остальных вершин. Тогда на каждом шаге алгоритма следующей нумеруется та вершина подмножества B, которая ведет к

присоединению наименьшего количества вершин из C к множеству B. После того как очередная вершина выбрана, производится соответствующее переопределение множеств A, B и C. Если граф не является связным, то алгоритм можно применять отдельно к каждой из его связных компонент.

Алгоритм KING учитывает тот факт, что часть вершин уже пронумерована – и связи этих вершин с остальными не скажутся на профиле и их можно не учитывать. Обращение нумерации KING не приводит к уменьшению профиля.

В приложении 1 приведен программный код на языке C# алгоритма KING.

2.4 УПОРЯДОЧЕНИЕ СЛОУНА

Алгоритм Слоуна (SLOAN) [5] основан на особом расчете приоритетов у вершин. Алгоритм на каждом шаге разбивает все множество вершин на 4 группы: пронумерованные (postactive); активные (active) и близкие к активным (preactive) вершины – эти вершины смежны пронумерованным вершинам и из них выбираются кандидаты для нумерации, вершины смежные активным вершинам (preactive); остальные неактивные вершины (inactive).

Алгоритм начинает работу с поиска псевдодиаметра и, соответственно, пары псевдопериферийных вершин, удаленных друг от друга на расстояние псевдодиаметра. Одну из этих вершин алгоритм принимает за стартовую, другую считает конечной. Для всех вершин графа алгоритм рассчитывает их числовые приоритеты. Значение приоритета вершины зависят от локальной степени и ее удаленности от конечной вершины. На первом шаге алгоритм относит конечную вершину группу preactive. Далее в цикле: в группах active и preactive выбирает вершину с наибольшим приоритетом, нумерует ее (эта вершина переносится в группу postactive) и корректирует приоритеты и группы всех смежных вершин (preactive становится active, inactive становиться preactive).

В приложении 1 приведен программный код на языке C# алгоритма SLOAN.

2.5 ИТЕРАЦИОННЫЕ УЛУЧШАЮЩИЕ СХЕМЫ

Итерационные методы минимизации профиля заключается в локальном обмене соседних строк/столбцом матрицы, приводящие к минимизации ленты и профиля. Как правило, эти методы оказываются неэффективными, если использовать их независимо от других подходов, однако могут давать неплохие результаты, если доступно хорошее начальное приближение к искомому упорядочению. Таким образом, улучшающие методы можно применять для улучшения упорядочений, полученных каким-либо другим методом.

3 МНОГОУРОВНЕВЫЙ АЛГОРИТМ

Ключевая идея многоуровневого алгоритма [6] состоит в следующем: размерность задачи редуцируется путем удаления наименее существенной информации, затем производится поиск решения редуцированной задачи, после этого производится ряд улучшений полученного решения на основе добавления в модель задачи ранее удаленной информации.

Важным достоинством многоуровневых алгоритмов являются широкие возможности по их настройке, поскольку они позволяют достаточно легко интегрировать в структуру алгоритма различные эвристики, направленные на улучшение получаемого решения.

3.1 ОБЩАЯ СТРУКТУРА АЛГОРИТМА

Многоуровневый алгоритм (МА) минимизации ленты и профиля имеет ряд отличий от классической многоуровневой схемы работы. Алгоритм на вход получает не только исходные данные задачи (граф), но и одно из допустимых решений (вариант упорядочения). Цель алгоритмы – исследование областей вокруг начального решения и поиск лучшего решения.

Процесс работы МА включает три фазы. На первой фазе (фаза загрубления) алгоритм осуществляет последовательную редукцию задачи (загрубление задачи) и начального решения (загрубление решения). Каждое загрубление уменьшает (редуцирует) размерность исходной задачи. Идеально, если алгоритм редукции на просто формирует задачу меньшей сложности, но такую, при которой хорошие решения загрубленной задачи соответствуют хорошим решениям исходной задачи. Процесс редукции продолжается не более определенного числа раз (предельный уровень редукции). Процесс может прерываться в случае нехватки оперативной памяти, бесперспективности редукции или по ограничению МА по времени работы.

Вторая фаза – фаза поиска начального решения самой загрубленной задачи. МА на этом этапе применяет две стратегии поиска лучшего решения. Первая стратегия заключается в локальном улучшении загрубленного начального решения. Вторая стратегия предполагает поиск нового решения загрубленной задачи, например, RCM алгоритмом с последующим локальным улучшением. В завершении фазы поиска из двух найденных решений МА выбирает лучшее с точки зрения ширины ленты.

Третья фаза – фаза восстановления решения. Здесь МА последовательно переходит от задачи более редуцированной к задаче менее редуцированной. На каждом переходе алгоритм восстанавливает и улучшает решение более редуцированной задачи. В этом момент МА владеет парой решений менее редуцированной задачи: начальное загрубленное и улучшенное решение и найденное восстановленное и улучшенное решение. МА выбирает лучшее с точки зрения ширины ленты и переходит к следующей менее редуцированной задачи.

Описанную выше схему можно запускать в итерационном режиме – начальное решение улучшается МА, найденное лучшее решение становиться начальным для следующей итерации МА. Итерационный процесс продолжается не более определенного числа раз (максимальное число итераций), и может прерываться в случае бесперспективности итерации или в связи с нехваткой времени работы алгоритма. Описанную схему работы МА будем называть многоуровневым итерационным алгоритмом (МИА).

Для определенности конкретный вариант программной реализации МИА, представленный в приложении 1, будем обозначать MLI. Также MLI будем обозначать упорядочение, полученное в результате работы этого алгоритма

3.2 ПРОЦЕДУРЫ РЕДУКЦИИ ЗАДАЧИ И ВОССТАНОВЛЕНИЯ РЕШЕНИЯ

Процедура редукции задачи является одной из самых важных в общей структуре МА. Качества работы редукции во многом определяет качество алгоритма в целом. Как уже отмечалось выше, алгоритм должен не только уменьшать размерность задачи, но переносить особенности ландшафта исходной задачи на ее редукцию.

Для задачи минимизации ленты предлагается алгоритм редукции, который группирует вершины и объединят вершины одной групп в супервершины редукции исходного графа. Тем самым граф-редукция становиться меньше исходного графа, а задача упрощается. Возникает вопрос, по какому принципу группировать вершины?

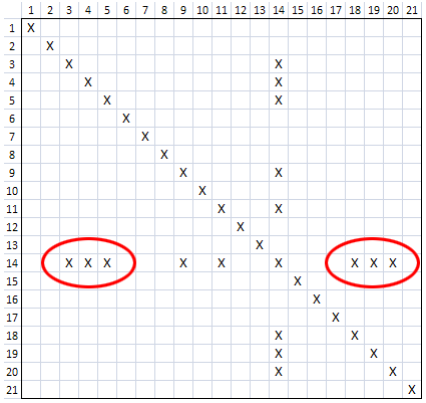


Рис.3. В строке выделены два крайних кластера
(группы подряд идущих ненулевых значений).

Для ответа на этот вопрос обратимся к рисунку 3, где для одной строки выделены две группы подряд идущих ненулевых значений. Подобные группы, в которые входит более одного ненулевого значения, будем называть кластерами. Особый интерес представляют крайние кластеры. Крайним кластером будем называть кластер такой, что либо справа, либо слева от кластера в строке нет ненулевых значений. Очевидно, любая строка может иметь не более чем 2 кластера (левый и правый). Главной особенностью крайнего левого кластера является то, что любые его перемещения в строке

(соответственно, симметричные перемещения в столбцах) обязательно повлияют на локальную ленту строки. Главной особенностью крайнего правого кластера является то, что любые его перемещения в строке обязательно повлияют на локальный фронт строки.

Алгоритм редукции осуществляет поиск крайних кластеров в упорядоченной матрице смежности графа и группирует вершины, которые соответствуют позициям кластера. Указанный алгоритм редукции будем именовать RIMP, вариант программной реализации алгоритма приведен в приложении 1.

3.3 ЛОКАЛЬНАЯ МИНИМИЗАЦИЯ ЛЕНТЫ

В процедуре локальной минимизации ленты используется простая техника локальных обменов соседних строк/столбцов матрицы смежности графа, приводящие к минимизации ленты и профиля. Указанная итерационная процедура работает до тех пор, пока есть возможность совершить успешный обмен. Как показывает практика, подобная полная локальная оптимизация может приводить к бесконтрольному росту вычислительных затрат МИА. Поэтому целесообразен параметр «предельное время работы», который ограничивает время работы процедуры локальной минимизации ленты.

3.4 ПАРАМЕТРЫ МНОГОУРОВНЕВОГО АЛГОРИТМА

Можно выделить следующие важные параметрами МИА: предельное время работы МИА, предельная глубина редукции (уровень редукции), предельное число итераций (число итераций), время работы процедуры локальной минимизации ленты (продолжительность локальной оптимизации).

ЛИТЕРАТУРА

1. Игнатьев А.В., Ромашкин В.Н. Анализ эффективности методов решения больших разреженных СЛАУ//Интернет-вестник ВолгГАСУ, Сер.: Строит. Информатика, 2008, вып. 3(6) стр. 1-7

2. Джордж А, Лю Дж. Численное решение больших разреженных систем уравнений. Пер. с англ. — М.: Мир, 1984. — 333 с.

3. Писсанецки С. Технология разреженных матриц. Пер. с англ. — М.: Мир, 1988. — 410с.

4. King L.P. An automatic reordering scheme for simultaneous equations derived from network problems // International journal for numerical methods in engineering, vol. 2, 1970, 523—533.

5. Sloan S.W. A fortran program for profile and wavefront reduction // International journal for numerical methods in engineering, vol. 28, 1989, 2651-2679.

6. Батищев Д.И., Старостин Н.В., Филимонов А.В. Многоуровневая декомпозиция гиперграфовых структур. Приложение к журналу «Информационные технологии» №5 (141) 2008, стр.1 – 32.

7. Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые задачи. — М.Мир, 1982. – 416с.

Николай Владимирович **Старостин**

МОДЕЛИ И МЕТОДЫ УПОРЯДОЧЕНИЯ РАЗРЕЖЕННЫХ МАТРИЦ

Учебно-методическое пособие

Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского»

603950, Нижний Новгород, пр. Гагарина, 23.