УДК 519.6; 519.67; 519.683

# АЛГЕБРО-ГЕОМЕТРИЧЕСКИЕ И ИНФОРМАЦИОННЫЕ СТРУКТУРЫ МЕТОДОВ ДЕКОМПОЗИЦИИ ОБЛАСТЕЙ

**Я. Л. Гурьева[1], В. П. Ильин[2], Д. В. Перевозкин[3]**

Рассматриваются алгебраические, геометрические и информационные аспекты параллельных методов декомпозиции для решения больших систем линейных уравнений с разреженными матрицами, возникающими при аппроксимации многомерных краевых задач на неструктурированных сетках. Алгоритмы базируются на разбиении сеточной расчетной области на подобласти с параметризованной величиной пересечений и различными интерфейсными условиями на смежных границах. Рассматриваются вопросы, возникающие при алгебраической декомпозиции исходной матрицы. Применяются различные двухуровневые итерационные процессы, включающие в себя предобусловленные крыловские методы с использованием грубосеточной коррекции, а также синхронное решение вспомогательных систем в подобластях с помощью прямых или итерационных алгоритмов. Распараллеливание алгоритмов реализуется средствами гибридного программирования с формированием MPI-процессов для каждой подобласти и использованием в них многопотоковых вычислений над общей памятью. Информационные коммуникации между соседними подобластями осуществляются на каждой внешней итерации путем предварительной организации буферов обмена и применения неблокирующих операций с возможностями проведения арифметических действий на фоне передачи данных.

**Ключевые слова:** декомпозиция областей, большие системы линейных уравнений, разреженные матрицы, структуры данных, гибридное программирование, параллельное программирование.

**1. Введение.** Целью настоящего анализа является описание и построение алгебро-геометрической структуры для больших сеточных систем линейных алгебраических уравнений (СЛАУ) с разреженными матрицами вида

$$Au = f, \quad A = \{a_{l,m}\} \in \mathcal{R}^{N,N}, \quad u = \{u_l\}, \quad f = \{f_l\} \in \mathcal{R}^N, \tag{1}$$

решение которых предполагается осуществлять с помощью параллельных методов декомпозиции областей (МДО) на многопроцессорных вычислительных системах (МВС) с общей и распределенной памятью, где расчеты реализуются с использованием многопотоковых вычислений (multi-thread computing) и MPI-процессов в системах передачи сообщений (Message Passing Interface) [1]. Кроме того, будут рассмотрены технологические и информационные вопросы, возникающие при программной реализации МДО на МВС и влияющие на производительность кода.

Под "большими" СЛАУ понимаются такие, объем данных для которых превышает размер оперативной памяти одного MPI-процесса. Понятие "сеточные алгебраические системы" означает, что последние получены из методов конечно-объемных или конечно-элементных аппроксимаций исходной функциональной многомерной (конкретно рассматриваются размерности $d = 2$ или $d = 3$) начально-краевой задачи на неструктурированной квазиравномерной сетке. Формально исходную непрерывную задачу запишем в виде

$$\begin{aligned} &Lu(\boldsymbol{r}) = f(\boldsymbol{r}), \quad \boldsymbol{r} \in \Omega, \\ &lu\big|_\Gamma = g(\boldsymbol{r}), \quad \boldsymbol{r} \in \Gamma, \quad \overline{\Omega} = \Omega \bigcup \Gamma, \end{aligned} \tag{2}$$

[1] Институт вычислительной математики и математической геофизики СО РАН, просп. Лаврентьева 6, 630090, Новосибирск; ст. науч. сотр., e-mail: yana@lapasrv.sscc.ru

[2] Институт вычислительной математики и математической геофизики СО РАН, просп. Лаврентьева 6, 630090, Новосибирск; гл. науч. сотр., e-mail: ilin@sscc.ru

[3] Институт вычислительной математики и математической геофизики СО РАН, просп. Лаврентьева 6, 630090, Новосибирск; мл. науч. сотр., e-mail: dperevozkin@mail.ru

где $L$ — дифференциальный оператор, $l$ — оператор краевых условий, а $f$ и $g$ — некоторые заданные функции. Краевые условия на разных частях границы будем различать двух видов: условия без производных (1-го рода, или Дирихле) и условия, содержащие пространственные производные (2-го и 3-го рода, т.е. Неймана и Робина соответственно). Если исходная задача является нелинейной, то полагается, что она решается с помощью какого-то итерационного метода линеаризации, на каждом шаге которого необходимо решать линейную вспомогательную задачу. И в данном случае рассматривается именно такой шаг. Отметим еще, что если задача (2) решается во временно́й области, т.е. является нестационарной и содержит производные по времени, то считается, что последние аппроксимируются неявным образом, так что на каждом временно́м шаге формируется СЛАУ вида (1). Таким образом, эффекты нелинейности и нестационарности ниже рассматриваться не будут. Предполагается, что исходные данные задачи (2) гарантируют существование единственного и достаточно гладкого решения, обеспечивающего применимость рассматриваемых численных методов.

Сетка, конструируемая для численного решения краевой задачи, а точнее — сеточная расчетная замкнутая область $\overline{\Omega}^h = \Omega^h \cup \Gamma^h$, представляет собой дискретизацию непрерывной расчетной открытой области $\Omega$ с границей $\Gamma$. Сетка определяется как совокупность своих элементов, или примитивов: узлов, ребер, граней и конечных объемов (в случае $2D$ грани являются конечными (плоскими) объемами), количество которых обозначаем через $N_n$, $N_e$, $N_f$ и $N_v$ соответственно. Длины ребер называются шагами сетки и обозначаются буквой $h$. Если они одинаковые, то сетка называется равномерной. Реально решение исходных задач осуществляется на последовательности сгущающихся сеток $\Omega^{(s)}$, $s = 1, 2, \ldots$, имеющих соответствующие характерные шаги $h^{(s)} \xrightarrow[s\to\infty]{} 0$. Последовательность сеток называется квазиравномерной, если при $s \to \infty$ отношение максимального шага сетки к минимальному остается конечным, т.е. $h^{(s)}_{\max}/h^{(s)}_{\min} = O(1)$. Конкретная сетка с конечным шагом $h^{(s)}$ называется квазиравномерной, если эта величина относительно невелика. Далее индекс "$s$" по контексту отбрасывается. Количество сеточных элементов оценивается по порядку как $N_n$, $N_e = O\big(h^{-d}\big)$, $N_f = O\left(h^{-(d-1)}\right)$, $N_v = O\big(h^{-d}\big)$. Сетка называется неструктурированной, если при любой нумерации ее элементов связи (инцидентности) между элементами можно указать только их перечислением.

Квадратная вещественная матрица $A = \{a_{l,m}\} \in \mathcal{R}^{N,N}$ порядка $N \sim h^{-d}$ (только такие мы и будем рассматривать) называется разреженной, если число ее ненулевых элементов в каждой строке много меньше порядка ($M_l \ll N$). Как правило, мы предполагаем, что $M_l$ не зависит от $h$ и что портрет матрицы (топология, т.е. взаимное расположение, или же совокупность индексов ее ненулевых элементов) является симметричным.

Совокупность номеров столбцов для ненулевых элементов $l$-й строки матрицы $A$ будем обозначать через $\omega_l$ и называть $l$-м строчным шаблоном, а также сеточным шаблоном, т.е. совокупностью номеров сеточных элементов, участвующих в $l$-м сеточном уравнении.

Понятие "сеточные СЛАУ" подразумевает, что каждое уравнение системы соответствует какому-то элементу сетки. В простейшем случае $l$-е уравнение (а также соответствующие компоненты векторов $u_l$, $f_l$) относятся к $l$-му узлу сетки. Такие уравнения называются узловыми, а в общем случае типы уравнений и/или неизвестных могут быть еще реберными, граневыми и объемными, в том числе одному сеточному элементу любого типа может соответствовать несколько векторных компонент (а также и уравнений, такие ситуации возникают при использовании базисных функций МКЭ высших порядков и при аппроксимации уравнений с неизвестными векторными функциями, см. [2] и цитируемую там литературу). Типичный пример — дискретная система двумерных уравнений Навье–Стокса, в которых неизвестные скорости относятся к серединам сеточных ребер, а давление — к серединам сеточных объемов.

Важным моментом для программирования больших разреженных неструктурированных СЛАУ является тот факт, что в матрице системы хранятся только ненулевые элементы каждой строки, а также номера столбцов, к которым они относятся. Это так называемый "сжатый разреженный строчный формат", или CSR (Compressed Sparse Row). Возможны и другие аналогичные сжатые форматы, но всех их объединяет повышенная сложность программирования и пониженная производительность кода ввиду непрямого пути доступа к данным.

Концепция рассматриваемых алгоритмов заключается в разбиении исходной задачи на подзадачи, которое можно делать на непрерывном (функциональном), сеточном (в том числе в терминах графов) и алгебраическом (векторно-матричном) уровнях. Структурно МДО представляет собой двухуровневый итерационный процесс, основанный на блочных предобусловленных алгоритмах в подпространствах Крылова, конструируемых с использованием исходной (густой) сетки и макросетки, составленных из подобластей. Реализация параллельных методов на МВС в идеале осуществляется без программных ограничений на число степеней

свободы задачи и количество используемых вычислительных устройств. Многообразие возникающих при этом вопросов ставит проблему системного подхода к построению математического и программного обеспечения для широкого круга актуальных прикладных задач.

Настоящая статья построена следующим образом. В разделе 2 описываются алгебро-геометрические аспекты методов декомпозиции сеточных расчетных областей. Раздел 3 посвящен описанию информационных структур и технологической схемы реализации рассматриваемых двухуровневых итерационных процессов, а в разделе 4 исследуются вопросы производительности параллельных МДО при их гибридном программировании на кластерных МВС. В заключении обсуждаются перспективы создания высокопроизводительного вычислительного инструментария для параллельных алгоритмов решения больших разреженных СЛАУ.

**2. Алгебро-геометрические аспекты МДО.** Рассмотренные выше понятия относятся к "стандартной" модели сетки, т.е. на "микроуровне". Решение больших СЛАУ с помощью параллельных МДО (или DDM — от Domain Decomposition Methods, см. [3–8] и цитируемые там работы) основано на представлении расчетной области (на непрерывном или сеточном уровне) совокупностью подобластей, образующих сетку "макроуровня", включающую в себя такие объекты, как макроузлы (вершины подобластей), макроребра, макрограни и макрообъемы (последние представляют собой подобласти).

Рассмотрим сеточную расчетную область $\Omega$ с границей $\Gamma$, разделенную на $P$ непересекающихся (non-overlapping) сеточных подобластей $\Omega_q \subset \Omega$ с границами $\Gamma_q$ (здесь и далее индексы "$h$" у сеточных объектов опускаем):

$$\overline{\Omega} = \bigcup_{q=1}^{P} \overline{\Omega}_q, \quad \overline{\Omega} = \Omega \bigcup \Gamma, \quad \overline{\Omega}_q = \Omega_q \bigcup \Gamma_q, \tag{3}$$

причем $\Omega_{q'} \bigcap \Omega_{q''} = 0$ при $q' \neq q''$. Для $q$-й подобласти обозначим через $\widehat{\omega}_q$ совокупность номеров соседних, или смежных, подобластей, имеющих общую часть границы с $\overline{\Omega}_q$. Границу каждой из подобластей можно представить суммой из $M_q$ ее частей ($M_q$ — общее количество подобластей, смежных с $\Omega_q$), по их принадлежности к соседним подобластям:

$$\Gamma_q = \bigcup_{r \in \widehat{\omega}_q} \Gamma_{q,r} \equiv \bigcup_{q'=1}^{M_q} \Gamma_{q,q'}, \quad \Gamma_{q,r} = \overline{\Omega}_q \bigcap \overline{\Omega}_{q'}. \tag{4}$$

Расчетные узлы могут лежать как в некоторой подобласти $\Omega_q$, так и на ее границе $\Gamma_q$.

При упоминании подобластей вида (3) под $\Omega_0$ (если это необходимо) мы формально всегда будем понимать "внешнюю" подобласть, т.е. дополнение расчетной области до всего пространства: $\Omega_0 = \mathcal{R}^d/\Omega$. В каждой подобласти будем выделять фрагменты границы $\Gamma_{q,r'}$ с индексами $r' = 0$ и $r' \neq 0$, составляющие соответственно внешнюю и внутреннюю границы: $\Gamma_q^0 = \bigcup \Gamma_{q,0}$ и $\Gamma_q^i = \Gamma_q / \Gamma_q^0$. Подобласти, у которых часть границы является внешней, называются околограничными, а остальные — внутренними.

Отметим, что для узлов из $\Gamma_{q,0}$ с граничными условиями 2-го или 3-го рода (совокупность таких узлов обозначим через $\Gamma'_{q,0}$) на этапе аппроксимации исходной задачи (2) формируется "обычное" сеточное уравнение

$$a_{l,l} u_l + \sum_{m \in \omega_l} a_{l,m} u_m = f_l, \tag{5}$$

где $\omega_l \in \Omega_q$ — сеточный шаблон для узла с номером $l$, т.е. фактически такие узлы из $\Gamma_{q,0}$ в СЛАУ никак не выделяются. С другой стороны, если граничный узел является "дирихлевым" (множество таких узлов обозначаем $\Gamma''_{q,0}$, причем $\Gamma_{q,0} = \Gamma'_{q,0} \bigcup \Gamma''_{q,0}$), то соответствующее сеточное уравнение имеет тривиальный вид $u_l = \text{const}$. В таких случаях практически всегда делается (и мы всегда эту процедуру предполагаем, хотя она, вообще говоря, не является обязательной) исключение условий Дирихле (constraining): значение $u_l$ исключается из всех соседних узлов, у которых корректируется правая часть и затем зануляется соответствующий коэффициент, т.е. при $m \in \omega_l$ и $l \in \Gamma''_{q,0}$ имеем:

$$f_m := f_m - a_{m,l} u_l, \quad a_{m,l} := 0.$$

Таким образом, узлы из $\Gamma_{q,0}$ при этом в СЛАУ (1) не входят.

Важно подчеркнуть, что формулы (3) и (4) определяют декомпозицию с явным определением внутренних сеточных границ между смежными подобластями. Совокупность таких границ-разделителей, не входящих ни

в одну открытую подобласть, образуют каркас, или скелет разделенных подобластей. Этот сеточный каркас, в свою очередь, можно тоже рассматривать как подобласть. Для определенности дадим ей последний $(P+1)$-й номер.

При построении разбиений (3) и (4) и формировании сеточных уравнений вида (5) вводится сплошная нумерация $1,\dots,N$ всех неизвестных и предполагается, что разбиению геометрических объектов на подобласти соответствует выделение подмножеств из введенного множества номеров.

Рассмотрим сначала сеточную макроструктуру данных для простейшего типа СЛАУ, когда все уравнения являются узловыми и каждому $l$-му узлу сетки соответствует одна векторная компонента $u_l$, $l=1,\dots,N$. В данном случае $l$ является глобальным номером узла и соответствующих векторных компонент $u_l$, $f_l$. Отметим, что при переупорядочивании узлов сетки и соответствующих строк матрицы $A$ ее структура меняется, но межузловые связи в сетке при этом сохраняются, т.е. они в некотором смысле являются инвариантом преобразования перестановок. Можно также сказать, что матричный портрет определяет некоторый граф, который является изоморфизмом по отношению к сеточной структуре. Для наглядной иллюстрации взаимосвязи матричных и сеточных структур мы рассмотрим модельную СЛАУ, представляющую собой простейшую семиточечную аппроксимацию задачи Дирихле для уравнения Пуассона в декартовой системе координат в кубической расчетной области на кубической же сетке с шагом $h=1/(M+1)$, для которой коэффициенты уравнений вида (5) при "естественной" глобальной нумерации узлов определяются следующим образом:

$$a_{l,l}=6,\quad a_{l,m}=-1,$$
$$l,m\in\omega_l=\bigl(l,l+1,l-1,l+M,l-M,l+M^2,l-M^2\bigr),$$

причем внедиагональные коэффициенты $a_{l,m}$ равны нулю, если $l$-й узел сетки в соответствующем координатном направлении примыкает к границе Г.

Пусть сетка имеет по $M$ узлов вдоль каждой координатной линии и $N=M^3$. Предполагаем, что макросетка из подобластей кубическая и $P=M_1^3$. С учетом наличия сеточного разделительного каркаса ($(P+1)$-й подобласти) справедливы соотношения

$$M=M_1L_1+M_1-1,\quad N_s=L_1^3,\quad s=1,\dots,P,$$
$$N_{P+1}=3L_1^2(M_1-1)+3L_1(M_1-1)+(M_1-1)^3,$$

где $N_s$ — число узлов в $s$-й подобласти. В примере на рис. 1 для двумерного случая координатные линии "обычной" сетки нарисованы тонкими линиями, а макросетки (каркаса) — толстыми, причем узлы подобласти-разделителя обозначены крестиками.

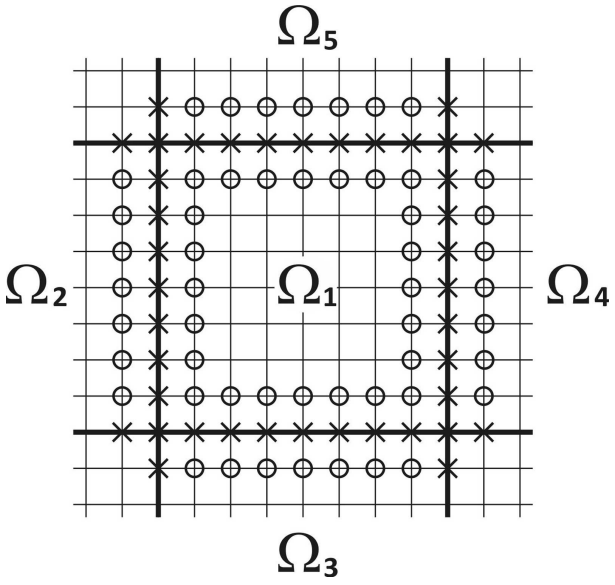


Рис. 1. Декомпозиции двумерной области с разделительной сеточной подобластью

Предположим, что глобальная нумерация узлов сеточной расчетной области, разбитой на $P$ непересекающихся подобластей, выбрана следующим образом: сначала идут все узлы 1-й подобласти, затем — 2-й, и так до $P$-й подобласти, а в заключение нумеруются узлы $(P+1)$-й подобласти, т.е. сеточного каркаса декомпозиции с разделителями. При соответствующей упорядоченности элементов СЛАУ ее матрица принимает блочный стрелообразный (стрела направлена из левого верхнего угла в правый нижний) вид:

$$A=\{A_{q,r}\}=\left|\begin{array}{ccc|c} A_{1,1} & & 0 & A_{1,P+1}\\ & \ddots & & \vdots\\ 0 & & A_{P,P} & A_{P,P+1}\\ \dots & \dots & \dots & \dots\\ \hline A_{P+1,1} & \dots & A_{P+1,P} & A_{P+1,P+1}\end{array}\right|. \qquad (6)$$

Другой возможный способ декомпозиции сеточной области — без формирования разделительных узлов. В этом случае можно представить, в отличие от определения $\Gamma_{q,q'} = \overline{\Omega}_q \bigcap \overline{\Omega}_{q'}$ из формулы (4), что геометрическая граница $\Gamma_{q,q'}$ проходит между узлами сетки: с одной стороны от нее находятся узлы $Q_l$ из $\Omega_q$, а с другой — из $\Omega_{q'}$. Формально это записывается следующим образом: $\Gamma_{q',q} = \bigcup\limits_l Q_l, l \in \omega_{l'}$, $l \in \Omega_q$, $l' \in \Omega_{q'}$. Соответствующий пример декомпозиции приведен на рис. 2.

Здесь напрашивается аналогия с интегральными граничными уравнениями для потенциалов простого и двойного слоя: в варианте с определением граничных разделителей на рис. 1 узлы из $\Gamma_{q,q'}$ образуют простой сеточный слой, а в альтернативном варианте без разделителей совокупность узлов из $\Gamma_{q,q'}$ и $\Gamma_{q',q}$ образует двойной сеточный слой.

В данном случае матрица $A$ при "естественной" (лексикографической) нумерации подобластей будет иметь уже не экзотический вид (6), а ленточную блочную форму (например, блочно-трехдиагональную или блочно-пятидиагональную).

Необходимо также отметить, что при построении "расширения" подобласти (в случае применения МДО с перекрытием подобластей) принципиальная разница между каркасным и бескаркасным делением области заключается в том, что при каркасном делении расширяться будет только сам каркас, таким образом "наезжая" на соседние подобласти (например, для случая на рис. 1 каркас будет включать в себя не только узлы, помеченные крестиками, но и узлы, помеченные кружочками), а при бескаркасном — расширяются все подобласти послойно (в смысле соседства узлов по аппроксимационному шаблону).

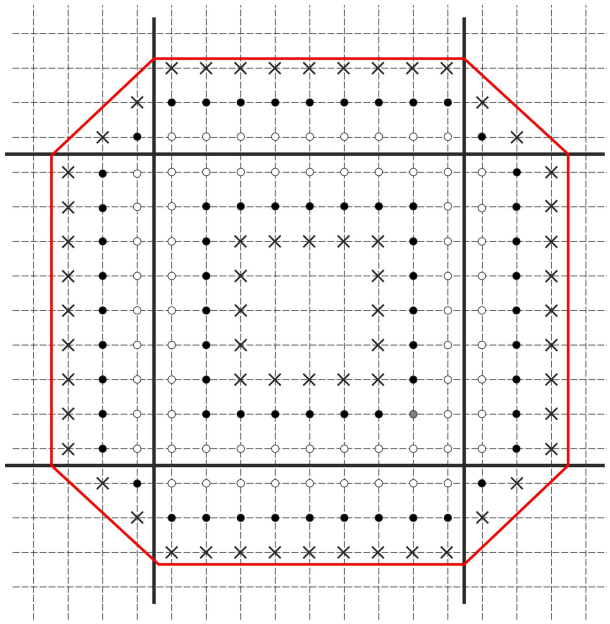

Рис. 2. Декомпозиция сеточной области без узлов-разделителей

При всей алгоритмической сложности и многообразии современных алгоритмов декомпозиции областей, основой параллельного решения рассматриваемых СЛАУ (в алгебраических терминах) является блочный итерационный метод Якоби, или другими словами (в геометрическом смысле) — аддитивный метод Шварца. Если обозначить через $\widehat{u}_q$, $\widehat{f}_q$, $q = 1, \ldots, P+1$, подвекторы, соответствующие подобласти $\Omega_q$, или соответствующей блочной строке матрицы $A$, то такой алгоритм записывается в виде

$$A_{q,q}\widehat{u}_q^n = f_q - \sum_{r \in \widehat{\omega}_q} A_{q,r}\widehat{u}_r^{n-1} \equiv \widehat{f}_q, \quad q = 1, \ldots, P; \quad n = 1, 2, \ldots, \tag{7}$$

где $n$ — номер итерации, а каждый диагональный блок $A_{q,q}$ — фактически матрица автономной вспомогательной СЛАУ для подобласти $\Omega_q$. Внедиагональные же блоки $A_{q,r}$, $r \neq q$, отвечают за взаимосвязи между подобластью $\Omega_q$ и контактирующими с ней подобластями, имеющими номера $r$ из множества $\widehat{\omega}_q$. Отметим, что введенные в (7) блоки $A_{q,r}$ в совокупности определяют блочную строку матрицы:

$$A_q = (\ldots \quad A_{q,r} \quad \ldots, r \in \widehat{\omega}_q), \quad A = \{A_q, q = 1, \ldots, P\}.$$

Итерационный процесс (7) легко формально обобщить, добавив к левой и правой частям формулы члены с некоторым одинаковым матричным множителем:

$$B_{q,q}\widehat{u}_q^n \equiv (A_{q,q} + C_{q,q})\widehat{u}_q^n = f_q + C_{q,q}\widehat{u}_q^{n-1} - \sum_{r \in \widehat{\omega}_q} A_{q,r}\widehat{u}_r^{n-1}, \quad n = 1, 2, \ldots. \tag{8}$$

Очевидно, что если при $n \to \infty$ итерационные процессы (7), (8) сходятся, то предельный вектор $u = \{u_q\}$ у них один и тот же, а именно — решение исходной СЛАУ. Вводимые дополнительные блоки $C_{q,q}$ не меняют

структуру и логику итерационного алгоритма, если они не изменяют портрет матрицы $B_{q,q}$ в сравнении с $A_{q,q}$. Обычно добавляются члены только к сеточным уравнениям для околограничных узлов из $\Omega_q$, и эта процедура соответствует изменению интерфейсных, или краевых, итерируемых условий между контактирующими подобластями. В частности, если положить $C_{q,q} = \theta D_{q,q}$, где $D_{q,q}$ — диагональная матрица, каждый элемент которой равен сумме внедиагональных элементов матриц $A_{q,r}$, $r \in \widehat{\omega_q}$, то при $\theta = 0$ интерфейс между смежными подобластями соответствует условию Дирихле, а при $\theta = 1$ — условию Неймана.

Подчеркнем, что мы для простоты рассматриваем до сих пор декомпозицию без пересечения подобластей, чтобы выделить главные алгебро-геометрические особенности алгоритмов. При наличии пересечений структура методов несколько усложняется. Например, блочное представление СЛАУ будет содержать перекрывающиеся блоки матрицы $A$ и подвекторов. Само построение пересечений целесообразно делать последовательно: сначала осуществляется конструирование подобластей $\Omega_q$ без пересечения, а затем каждая из подобластей постепенно расширяется путем включения в нее по одному сеточному слою на очередном этапе. Величину пересечений естественно измерять параметром $\Delta$ — числом расширяющих (дополнительных) сеточных слоев подобласти $\Omega_q$, который для простоты будем считать одинаковым для всех $q$. Последовательность формируемых границ можно описать следующим образом:

$$\begin{aligned} &\Gamma_q \equiv \Gamma_q^0 = \left\{ l' \in \omega_l, \quad l \in \Omega_q, \quad l' \notin \Omega_q, \quad \Omega_q^1 = \overline{\Omega}_q^0 = \Omega_q \bigcup \Gamma_q^0 \right\}, \\ &\Gamma_q^t = \left\{ l' \in \omega_l, \quad l \in \Omega_q^{t-1}, \quad l' \in \Omega_q^{t-1}, \quad \Omega_q^t = \overline{\Omega}_q^{t-1} = \Omega_q^{t-1} \bigcup \Gamma_q^{t-1} \right\}. \end{aligned} \tag{9}$$

В частности, на рис. 2 изображен пример декомпозиции области с пересечением контактирующих подобластей величиной в три сеточных слоя. А на рис. 1 кружочками обозначены узлы из $\Gamma_{P+1}^0$, граничные к подобласти-разделителю. В дальнейшем расширенную подобласть с числом расширяющих слоев $\Delta$ будем обозначать через $\Omega_q^\Delta$.

Для полноты описания МДО необходимо представить методы грубосеточной коррекции, имеющие в разных версиях названия алгоритмов дефляции, агрегации и основанные алгебраически на малоранговой аппроксимации исходной матрицы $A$ [4–7]. На качественном уровне данный подход заключается в построении грубой сетки с числом узлов $N_c \ll N$. Формально эта редкая сетка может быть никак не связана с макросеткой, составленной из подобластей, однако наиболее естественным является выбор $N_c = P$, когда каждой подобласти сопоставляется один узел. На построенной редкой сетке определяется некоторая система базисных финитных функций $\varphi_q(\boldsymbol{r})$, $q = 1, \ldots, P$, с помощью которой формируется (а затем решается) СЛАУ порядка $N_c$ относительно достаточно грубого галеркинского приближения к искомому вектору. Проще всего взять базисные функции нулевого порядка, т.е. кусочно-постоянные: $\varphi_q(\boldsymbol{r}) = 1$ при $\boldsymbol{r} \in \Omega_q$ и $\varphi_q(\boldsymbol{r}) = 0$ в противном случае. Тогда с их помощью определяется прямоугольная матрица полного ранга $W = \left(w^1 ... w^{N_c}\right) \in \mathcal{R}^{N,N_c}$, $w^q = \left\{ w_l^q = \varphi_q(\boldsymbol{r}_l),\, l \in \Omega^h \right\}$, компоненты вектор-столбцов которой — это значения базисных функций в узлах исходной сетки. Опуская детали (см. подробнее [6]), алгоритм можно представить с помощью "грубосеточной" предобусловливающей матрицы

$$B_c^{-1} = W \widehat{A}^{-1} W^T, \quad \widehat{A} = W^T A W \in \mathcal{R}^{N_c,N_c}, \tag{10}$$

простейшее применение которой заключается в построении "хорошего" начального приближения. В любом современном итерационном процессе в подпространствах Крылова вычисления начинаются с построения начального вектора $r^0$ и направляющего вектора $p^0$. Пусть $u^{-1}$ будет произвольный вектор. Тогда до начала итераций определяем величины

$$\begin{aligned} &u^0 = u^{-1} + B_c^{-1} r^{-1}, \quad r^{-1} = f - A u^{-1}, \\ &r^0 = f - A u^0, \quad p^0 = r^0 - B_c^{-1} r^0, \end{aligned} \tag{11}$$

что обеспечивает выполнение свойств ортогонализации

$$W^T r^0 = 0, \quad W^T A p^0 = 0, \tag{12}$$

по отношению к "пространству дефляции" $\mathrm{Span}\left\{ \mathrm{w}^1, \ldots, \mathrm{w}^{\mathrm{N_c}} \right\}$. Отметим, что формулы (11), (12) реализуются сравнительно экономно, поскольку требуют только решения вспомогательной "агрегированной" СЛАУ с матрицей $\widehat{A}$ малого порядка, которая в алгебраическом смысле представляет собой малоранговую аппроксимацию

исходной матрицы $A$. Требуемое решение можно получить с помощью прямого или некоторого итерационного метода.

В качестве примера итерационного процесса крыловского типа приведем мультипредобусловленный метод полусопряженных невязок (MPSCR, от Multi-Preconditioning Semi-Conjugate Residual, [6])

$$u^{n+1} = u^n + P_n\overline{\alpha}_n, \quad r^{n+1} = r^n - AP_n\overline{\alpha}_n, \quad n = 0, 1, \ldots,$$

где $P_n = \left\{ p_k^n = \left(B_n^{(k)}\right)^{-1} r^n,\ k = 1, \ldots, m_n \right\} \in \mathcal{R}^{N,m_n}$ — суть составленные из направляющих векторов матрицы, а $\overline{\alpha}_n = \left\{\alpha_k^n\right\} \in \mathcal{R}^{m_n}$ — векторы итерационных параметров, которые по условиям ортогональности

$$\left(Ap_k^n, Ap_{k'}^{n'}\right) = \rho_{n,k}\delta_{k,k'}\delta_{n,n'}, \quad \rho_{n,k} = \left(Ap_k^n, Ap_k^n\right), \tag{13}$$

определяются соотношениями

$$\alpha_k^n = \frac{\left(r^n, A\left(B_n^{(k)}\right)^{-1} r^n\right)}{\rho_{n,k}}, \quad k = 1, \ldots, m_n, \tag{14}$$

обеспечивающими минимум нормы невязки $\left\|r^{n+1}\right\| = \left(r^{n+1}, r^{n+1}\right)^{1/2}$ в блочном подпространстве Крылова $\mathcal{K}(\mathcal{A}, \mathcal{P}_\prime) = \mathrm{Span}\left\{\mathrm{P}_0, \mathrm{AP}_0, \ldots, \mathrm{A^nP}_0\right\}$. В этих формулах на каждой итерации участвует $m_n$ предобусловливателей $B_n^{(k)}$, которые могут меняться от шага к шагу.

Направляющие матрицы по условиям (13) определяются из "длинных рекурсий"

$$P_{n+1} = Q_{n+1} - \sum_{k=0}^{n}\sum_{l=0}^{m_k}\beta_{k,l}^n p_l^k, \quad Q_{n+1} = \left\{ q_k^{n+1} = \left(B_{n+1}^{(k)}\right)^{-1} r^{n+1} \right\},$$
$$\beta_{k,l}^n = \left(Ap_l^k, A\left(B_n^{(k)}\right)^{-1} r^n\right)\rho_{n,l}, \quad k = 1, \ldots, m_n.$$

В качестве предобусловливающих матриц $B_{n+1}^{(k)}$ на каждой итерации естественно брать блочно-диагональную матрицу $B_S = \text{block-diag}\{B_{q,q}\}$ из формулы аддитивного метода Шварца (8), а также дефляционную матрицу $B_c$ из (10), однако в последнем случае необходимо иметь в виду, что при большом количестве столбцов $N_c$ в матрице $W$ данная процедура является достаточно ресурсоемкой и применять ее целесообразно только периодически.

Недостатком рассмотренного алгоритма, как и других крыловских методов для решения несимметричных СЛАУ, является необходимость хранения всех направляющих векторов при большом числе итераций, что требует существенных затрат памяти. Для экономии памяти имеются два основных подхода.

Первый — это проведение рестартов через заданное число итераций $n_{\mathrm{rest}}$, когда очередной вектор невязки вычисляется не из рекурсий, а из уравнения, т.е. $r^{n_r} = f - Au^{n_r}$, а сам процесс начинается заново с полученного приближения $u^{n_r}$. Здесь $n_r = r \cdot n_{\mathrm{rest}}$, а $r = 1, 2, \ldots$ — номер рестарта. На каждом рестарте целесообразно делать грубосеточную коррекцию текущего приближения по формулам (11).

Второй способ заключается в ограниченной ортогональности, когда "старые" направляющие векторы отбрасываются, а хранятся и используются для расчетов только данные из последних $n_{\mathrm{lim}}$ итераций. Использование обоих приемов приводит к замедлению скорости сходимости итераций, но это есть неизбежная плата за экономию оперативной памяти.

При реализации метода декомпозиции с пересечением подобластей на каждой итерации возникает неоднозначность определения векторных компонент, которые находятся из решения двух или более подсистем. Для ее устранения существуют различные подходы, один из которых — "ограничительный" аддитивный метод Шварца (RAS — Restricted Additive Schwarz, см. [4]) — записывается следующим образом:

$$B_{\mathrm{RAS}}^{-1} = R\widehat{A}^{-1}W, \quad \widehat{A} = W^T A W = \text{ block-diag}\left\{\overline{A}_{q,q} \in \mathcal{R}^{\overline{N}_q,\overline{N}_q}\right\}.$$

Здесь $W = [w_1 \ldots w_P] \in \mathcal{R}^{N,P}$ — прямоугольная матрица, каждый столбец $w_q$ которой имеет единичные компоненты в узлах из $\Omega_q^\Delta$ и нулевые — в остальных, а матрица $R \in \mathcal{R}^{N,N}$ состоит из блочных строк $R_q \in$

$\mathcal{R}^{N_q,N}$, каждая из которых представляет собой оператор сужения $\Omega$ в $\overline{\Omega}_q$, т.е. $R_q u = u_q$ (напомним, что $\overline{\Omega}_q$ означает подобласть с числом узлов $N_q$ после декомпозиции без расширения, а $\Omega_q^\Delta$ — это расширенная подобласть). Отметим, что даже при симметричности исходной СЛАУ предобусловливающая матрица $B_{\mathrm{RAS}}$ в общем случае симметричной не является.

**3. Структуры данных и технологическая схема двухуровневых МДО.** Искусство программной реализации МДО на конкретной архитектуре МВС заключается, помимо минимизации межпроцессорных обменов, в сбалансированной декомпозиции областей, чтобы не допустить простоев отдельных вычислительных устройств. Здесь надо иметь в виду, что на каждой внешней итерации между контактирующими подобластями необходимо организовать экономичные коммуникации с возможно меньшими временны́ми затратами. С этой целью следует сформулировать задачу о создании некоторого достаточно универсального инструментария, который мог бы эффективно использоваться в различных программных разработках. Постановку проблемы в данном случае можно описать в некотором смысле как задание некоторого блочного CSR-формата.

Мы предполагаем, что значения последовательных приближений решения $u_l^n$ для $l \in \Omega_q$ находятся в своих "подобластях", т.е. в соответствующих MPI-процессах, причем они там располагаются в локальных упорядоченностях, настроенных на интерфейсные требования алгебраических решателей для вспомогательных СЛАУ в подобластях. При этом должны существовать процедуры перевода элементов $u_l^n$ из локальных упорядоченностей в глобальную и обратно. В данном случае рассматриваются простейшие интерфейсные связи типа "Дирихле–Дирихле", когда на каждой итерации происходят обмены вида граничных условий, не содержащих производных от решения.

Пусть известны следующие данные о декомпозиции сеточной расчетной области, совокупность которых мы обозначим как DD-CSR-формат:

— $N$, $P$ — общее количество узлов сетки $\Omega$ и число подобластей соответственно;

— $N_q$, $M_q$ — количество узлов в подобласти $\Omega_q$ и число контактирующих с ней подобластей, т.е. число элементов в $\widehat{\omega}_q$;

— $r_1, \ldots, r_{M_q}$ — номера контактирующих с $\Omega_q$ подобластей;

— $N^{(s)}_{q,r_1}, \ldots, N^{(s)}_{q,r_{M_q}}$ — размеры числовых массивов из $\Omega_q$, которые надо переслать (с помощью операции "send" в $q$-м MPI-процессе) в соседние подобласти с номерами $r \in \widehat{\omega}_q = \{r_1, \ldots, r_{M_q}\}$;

— $N^{(r)}_{r_1,q}, \ldots, N^{(r)}_{r_{M_q},q}$ — размеры вещественных массивов, которые принимаются (с помощью операции "receive") $q$-м MPI-процессом (подобластью) из соседних подобластей с номерами $r \in \widehat{\omega}_q$.

На последних двух типах данных уже фактически заканчивается аналогия между CSR и DD-CSR, поскольку первый формат описывает структуру скалярной матрицы $A = \{a_{l,m}\}$, а второй относится к ее блочному представлению $A = \{A_{q,r}\}$.

Фактически на каждой итерации в подобласть $\Omega_q$ требуется переслать из контактирующих подобластей $\Omega_r$, $r \in \widehat{\omega}_q$, значения $u_l^n$ в узлах, лежащих на участках ее границы $\Gamma_{q,r}$ и находящихся в соседних подобластях, или MPI-процессах. В целях сокращения времени пересылок соответствующие данные, расположенные хаотично, требуется предварительно собрать в цельный массив — буфер обмена. И наоборот, из подобласти $\Omega_q$ в соседние к ней подобласти $\Omega_r$ необходимо передать сеточные значения из узлов, находящихся на границах $\Gamma_{r,q} \in \Omega_q$.

Обозначим совокупность сеточных значений $u_l^n$ на $\Gamma_{q,r}$, по контексту, той же буквой $\Gamma_{q,r}$. Предполагаем, что эти величины расположены в вещественном массиве $\Omega_q$ при некоторой локальной упорядоченности узлов в данной подобласти, а их номера задаются простым перечислением в данной локальной нумерации. Соответствие между локальной и глобальной нумерацией устанавливается простым правилом: в вещественном массиве $u_l^n$ для $l \in \Omega$ сначала располагаются значения из подобласти $\Omega_1$, затем — из $\Omega_2$ и т.д. до $\Omega_{P+1}$. Соответствие глобального номера $l$ узла сетки его локальному номеру $l_q$ в подобласти $\Omega_q$ осуществляется с помощью простых формул:

$$l = l_q + N_q^b, \quad N_q^b = \sum_{q'=1}^{q-1} M_{q'}, \quad N_1^b = 0,$$

где $N_q^b$ — суммарное количество узлов (а в общем случае — неизвестных переменных) в подобластях, предшествующих $\Omega_q$.

Очевидно, что для доступа к $u_l^n$ необходимы целочисленные массивы (которые мы обозначим через $NS_{q,r}$, $r = 1, \ldots, M_q$) их локальных номеров в $\Omega_q$, имеющие по $S_{q,r}$ элементов в каждом. Необходимо иметь в виду, что для обеспечения экономичных коммуникаций по значениям всех $NS_{q,r}$ и $\Omega_q$ требуется еще сформировать вещественные массивы (обозначаемые через $AS_{q,r}$, $r = 1, \ldots, M_q$), содержащие подряд значения $u_l^n$ для $l \in NS_{q,r}$, которые являются буферами обменов между MPI-процессами для контактирующих подобластей, а точнее говоря — буферы для рассылки (send) из $\Omega_q$ в $\Omega_r$. Кроме того, необходимо еще описать буферы обратных обменов приема (receive) в $\Omega_q$, т.е. в $q$-м MPI-процессе, значений $u_l^n$ из соседних подобластей для $l \in \Omega_r$, $r \in \widehat{\omega}_q$. Это требует определение еще $M_q$ целых и столько же вещественных массивов, причем каждая такая пара имеет по $R_{r,q}$ элементов (эти количества совсем не обязаны совпадать с $S_{q,r}$); вещественные массивы предназначены для хранения $u_l^n$ после их приема из контактирующих с $\Omega_q$ подобластей, а целые величины — это их локальные номера в $\Omega_q$, куда их надо разослать из соответствующих буферов приема данных. Таким образом, структура данных для каждой из $q$-х подобластей должна содержать следующую информацию:

— $\Omega_q$ — вещественный массив длиной $N_q$ со значениями $u_l^n$ при $l \in \Omega_q$;

— $NS_{q,r}$ — целочисленные массивы с длинами $N_{q,r}$, $r = r_1, \ldots, r_{M_q}$, содержащие локальные номера $l_q$ в массиве $\Omega_q$, из которых надо сформировать буферы передачи данных $AS_{q,r}$ из $\Omega_q$ в $\Omega_r$;

— $AS_{q,r}$ — вещественные массивы со значениями $u_l^n$ — указанные выше буферы для рассылки из $\Omega_q$ в $\Omega_r$, длины и количества этих массивов — как в $NS_{q,r}$;

— $AR_{r,q}$ — аналогичные $AS_{q,r}$ вещественные массивы, но только являющиеся буферами приема данных в $\Omega_q$ из соседних подобластей $\Omega_r$, $r \in \widehat{\omega}_q$; количество таких массивов равно $M_q$, а длина каждого из них — $R_{r,q}$;

— $NR_{r,q}$ — аналогичные $AR_{r,q}$ по длинам и по их количеству массивы, но только целочисленные и содержащие локальные номера в $\Omega_q$ соответствующих значений из массивов $AR_{r,q}$.

В целом вычислительно-информационные операции двухуровневого МДО можно представить следующим образом. На каждой $n$-й внешней итерации обмены между MPI-процессом, соответствующим подобласти $\Omega_q$, и всеми остальными выполняются в три стадии:

— формирование из $u_l^n$, $l \in \Omega_q$, буферов пересылки данных $AS_{q,r}$ во все подобласти $\Omega_r$, $r \in \omega_q$ (их количество равно $M_q$);

— отправка сформированных буферов по соседним подобластям и прием от них буферов $AR_{q,r}$;

— вычисление правых частей вида (7) или (8) во всех подобластях $\Omega_q$.

Заметим, что для выполнения всех вышеперечисленных операций следует использовать неблокирующие функции `MPI_Isend` и `MPI_Irecv`. Это позволяет инициировать обмены в любом порядке, в то время как использование `MPI_Send` и `MPI_Recv` требует строгого определения последовательности сообщений, усложняя логику программы. Не менее важным фактором является возможность совмещать обработку и передачу данных, что приводит к повышению производительности вычислений.

Общая технологическая схема построения параллельного двухуровневого МДО достаточно простая: внешние итерации по подобластям выполняются средствами системы пересылки данных MPI, с помощью которой осуществляются встречные коммуникации между подобластями $\Omega_q$, для каждой из которых формируется свой MPI-процесс, включающий в себя автономное (и, как следствие, одновременное по всем подобластям) решение $q$-й подсистемы в подобласти на выделяемом для этого многоядерном процессоре с помощью различных прямых или итерационных процедур. Именно на последнем факторе и основывается распараллеливание МДО на МВС. Нижний уровень вычислительного процесса распараллеливается с помощью реализуемых системой OpenMP многопотоковых (multi-thread) вычислений над общей памятью. Именно эффективное гибридное (MPI + Open MP) распараллеливание должно обеспечить суммарную высокую производительность.

**4. Особенности параллельной реализации МДО.** При построении параллельных версий методов декомпозиции областей и рассмотрении проблем их оптимизаций необходимо иметь в виду два главных противоречивых фактора: математическая эффективность алгоритмов и производительность их программных реализаций на кластерной архитектуре МВС. Как правило, стремление к повышению скорости сходимости итераций и сокращению объема арифметических операций приводит к усложнению логической сложности алгоритмов и дополнительным затруднениям для достижения масштабируемого параллелизма на конкретной компьютерной платформе.

Здесь необходимо хотя бы коротко остановиться на принципах масштабируемого параллелизма в слабом и сильном смыслах. Первый подразумевает, что при одновременном пропорциональном росте числа степеней свободы задачи и количества используемых вычислительных устройств время ее решения остается примерно постоянным, а второй — при фиксированной размерности СЛАУ с ростом количества процессоров $P$ время решения убывает как $1/P$. Важно отметить, что в идеальном случае реализация алгоритмов на МВС должна осуществляться без программных ограничений на число степеней свободы и на количество вычислительных устройств. С точки зрения масштабируемого распараллеливания МДО это означает, что при $N \to \infty$ мы должны предусматривать возможность предела $P \to \infty$.

Главной задачей распараллеливания алгоритмов является обеспечение высокопроизводительных вычислений (HPC — High Perfomance Computing), которые зависят, в свою очередь, от сокращения коммуникационных потерь, т.е. обмена данными между устройствами памяти. Последние являются "вредными" не только с той точки зрения, что операции передачи информации на порядки медленнее арифметических. В последние годы, с появлением постпетафлопсных компьютеров, не менее важным становится то обстоятельство, что эти обмены — наиболее энергозатратные операции, являющиеся главными ограничительными (экономическими и техническими) препятствиями к росту мощностей суперкомпьютеров.

Для анализа эффективности распараллеливания алгоритмов необходимо иметь хотя бы грубую математическую модель МВС, которые на текущем этапе своего развития представляют собой слишком сложные и большие системы, бесконечно далекие от машины Тьюринга, чтобы попытаться исследовать вычислительно-информационные процессы аналитически.

Мы остановимся на следующей простейшей модели вычислений на типовом кластере, для которой и будем проводить анализ МДО. Предполагается, что МВС представляет собой сеть многопроцессорных узлов (с распределенной по ним памятью — distributed memory), содержащих по несколько многоядерных центральных и графических вычислительных устройств (CPU и GPGPU — Central Processor Unit и General Purpose Graphic Processor Unit), каждое из которых имеет свою общую многоуровневую иерархическую память (shared memory).

В идеале решение задачи автоматизации и оптимизации распараллеливания алгоритмов хотелось бы искать путем имитационного моделирования компьютерной системы в целом, однако эта проблема слишком сложна, и в данном случае приходится вовлекать полуэмпирические приемы или простейшие модели машинных вычислений. Примерами характеристик распараллеливания могут служить две величины — коэффициенты ускорения вычислений и эффективности использования процессоров:

$$S_p = T_1/T_p, \quad E_p = S_p/P, \tag{15}$$

где $T_p$ — время выполнения задачи или алгоритма на $P$ процессорах. Идеальной ситуацией является такая, когда значение $S_p$ прямо пропорционально $P$, а $E_p = 1$, но на практике зачастую придется довольствоваться коэффициентами эффективности в несколько процентов.

Главная цель при программировании параллельных алгоритмов заключается в минимизации информационных обменов, поскольку общее время решения задачи $T$ определяется двумя величинами:

$$T_a = N_a\tau_a, \quad T_c = M(\tau_0 + N_c\tau_c), \tag{16}$$

где $\tau_a$ и $\tau_c$ — усредненные времена выполнения одной арифметической операции и передачи одного числа, $N_a$ — количество арифметических действий, $M$ — число обращений к памяти, $\tau_0$ — время задержки (настройки) одной обменной операции, а $N_c$ — объем одного передаваемого массива данных. При этом надо иметь в виду характерные соотношения $\tau_0 \gg \tau_c \gg \tau_a$.

По поводу приведенных выражений необходимо сделать ряд замечаний. Во-первых, межузловые передачи информации целесообразно делать как можно реже и большими порциями, формируя заранее буферы обмена

данными. Во-вторых, усредненная величина $\tau_a$ является слишком условной, так как времена выполнения различных арифметических операций могут сильно отличаться между собой, а также значительно изменяться в зависимости от операционной обстановки (возможности конвейеризации, разные уровни оптимизации кода, которые могут осуществляться компилятором или программистом, и т.д.). Здесь картина настолько сложна, с учетом особенностей многоуровневой памяти и арифметических устройств, что разбираться в ней принципиально нет смысла, поскольку достаточно простых оценок, существенно уточняющих (15), (16), не существует. В-третьих, реальные времена коммуникаций зависят от физической конфигурации вычислительных узлов, а программируются обмены только по их логическим номерам. При этом важный положительный момент заключается в возможности некоторого совмещения арифметических и коммуникационных операций, так что полное время решения задачи $T$ может быть существенно меньше суммы $T_a+T_c$. Однако средства управления такими процессами достаточно ограничены. Наиболее перспективными подходами к исследованию или даже оптимизации фактической производительности параллельных реализаций являются численные эксперименты с активным использованием существующих профилировщиков и других аналитических инструментариев, дающих апостериори динамику загрузки различных вычислительных устройств МВС.

Переходя непосредственно к параллельной реализации МДО, необходимо отметить, что в этом случае требуется на каждой итерации осуществлять следующие арифметические действия:

— умножение матрицы $A$ на вектор,

— построение правых частей для вспомогательных СЛАУ вида (7) или (8),

— решение вспомогательных СЛАУ с предобусловливающими матрицами $B$, или умножение векторов на обратные матрицы $B^{-1}$,

— векторные операции: сложение, вычисление скалярного произведения, масштабирование, т.е. умножение на скаляр.

Необходимо отметить, что для вычисления векторных норм и скалярных произведений вида (13)–(14) во внешнем итерационном процессе целесообразно использовать функцию `MPI_Allreduce`, которая, с одной стороны, позволяет получить искомое значение, имея частично вычисленные скалярные произведения по подобластям, а с другой стороны, является механизмом синхронизации работы параллельной программы, представленной набором `MPI`-процессов.

С точки зрения программирования вычислительного процесса нам было бы проще допустить, что все исходные данные по решаемой СЛАУ расположены изначально в одном (управляющем) `MPI`-процессе, который начинает реализацию алгоритма с распределения информационных блоков СЛАУ по всем остальным сформированным `MPI`-процессам. Однако такая тактика может существенно ограничить размер решаемых СЛАУ из-за того, что для обеспечения необходимой на практике точности СЛАУ должны решаться, как правило, со стандартной двойной точностью, предполагающей 8-байтовое представление вещественного числа с плавающей запятой, или точкой. Чтобы этого не произошло, такой MPI-процесс необходимо обеспечить дополнительной памятью. Другой вариант — рассматривать параллельную реализацию МДО в предположении, что все исходные данные для СЛАУ уже распределены по $P$ подобластям и по соответствующим `MPI`-процессам (желательно также, хотя формально это и не обязательно, чтобы каждому процессу было доступно несколько ядер с общей памятью). При этом в идеале формирование распределенной СЛАУ осуществлялось сразу на стадии сеточной аппроксимации исходной непрерывной задачи, причем параллельным образом. Однако возможна и компромиссная тактика с реализацией МДО в два этапа: на первом сначала формируется “глобальная” СЛАУ в общей памяти, затем она агрегируется в блоки и распределяется по своим MPI-процессам, а на втором этапе уже непосредственно выполняется итерационное решение по подобластям. В приведенных выше описаниях вычислительных методов и технологий явно присутствует дуализм принципов декомпозиции областей. С одной стороны, данный подход естественно излагается на геометрическом, или сеточном, уровне: подобласть есть множество узлов, для которых определены понятия расстояния, топологии, метрики и различные операции над пространственными объектами. В частности, изложение соответствующих алгоритмов наглядно иллюстрируется рисунками, которые зачастую помогают находить элегантные вычислительные решения. Для совокупности таких подходов сложилось название геометрическая декомпозиция.

Однако формально СЛАУ, решаемую с помощью МДО, можно также рассматривать как “вещь в себе”, абстрагируясь от истории ее сеточного происхождения и тем более от изначально поставленной на функциональном языке краевой задачи. В таком случае исходные данные представляются только матрицей, задаваемой в формате CSR или каким-либо другим способом. При этом отсутствие геометрической информации компенсируется за счет вещественных массивов значений матричных элементов и целочисленного описания матричного портрета, который в определенном смысле изоморфен сеточному графу (например, существует очевидная аналогия между понятиями сеточного шаблона и структурой соответствующей строки матрицы). Декомпозиция области на алгебраическом языке означает блочное разбиение матрицы, а итерационные аддитивные методы Шварца, как уже отмечалось, — это блочные алгоритмы Якоби. Таким образом, фактически можно говорить об алгебраической декомпозиции как о самостоятельном и замкнутом направлении исследований. Хотя, конечно, геометрическая и алгебраическая декомпозиции — это две стороны одной медали, и данные методологии должны естественным образом дополнять и обогащать друг друга.

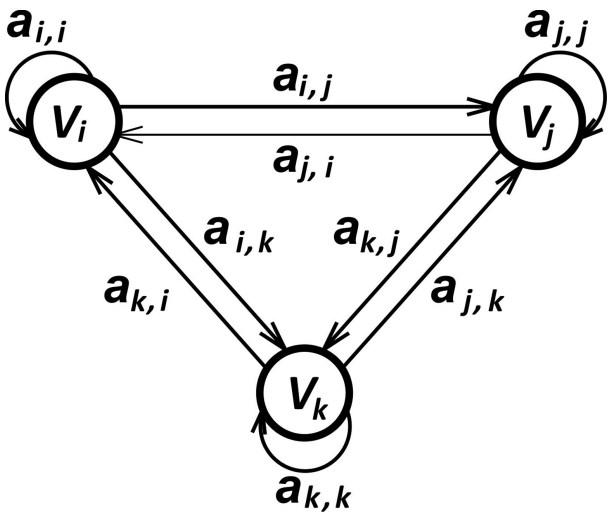


Рис. 3. Пример взвешенного ориентированного графа

В рассматриваемом случае разбиение матрицы на блоки естественным образом опирается только на матричный портрет. Необходимо иметь в виду, что конструктивным средством описания и анализа матрицы $A = \{a_{i,j}\}$ или другой дискретной системы является понятие графа. Напомним основные понятия из теории графов, которые требуются для построения программной реализации разбиения матрицы.

Граф $G = (V, E)$ определяется как совокупность вершин $V$ и ребер $E$ [9]:

$$V = \{v_i\}_{i=1}^{N}, \quad E = \big\{(v_i, v_j) | a_{i,j} \neq 0 \vee a_{j,i} \neq 0\big\}.$$

В общем случае рассматриваемые нами графы являются взвешенными и ориентированными, т.е. каждой паре вершин $(v_i, v_j)$ соответствует двойное ребро с весами $a_{i,j}$ и $a_{j,i}$, а отдельной вершине $v_k$ еще ребро-петля с “диагональным” весом $a_{k,k}$, см. рис. 3.

Если матрица $A$, или система сеточных СЛАУ, является симметричной, то достаточно рассматривать неориентированный граф с простыми ребрами и весами $a_{i,j} = a_{j,i}$. При описании же матричного портрета, или сеточного графа, введение весовых коэффициентов не требуется. В приложении к нашим задачам граф $V$ — это сеточная расчетная область $\Omega$, или множество строк матрицы $A$, или множество компонент вектора неизвестны $u = \{u_l, l \in \Omega\}$, а подмножество $W$ представляет собой какую-то из сеточных подобластей $\Omega_q$ или соответствующую блочную строку $A_q = \{A_{q,r}, r \in \widehat{\Omega}_q\}$ матрицы.

Важным для МДО понятием является понятие смежного подмножества: если $W$ — некоторое подмножество вершин графа $G$, то его смежным подмножеством $\mathrm{Adj}(W)$ называют совокупность всех вершин, смежных какой-либо вершине из $W$, но не принадлежащих $W$: $\mathrm{Adj}(W) = \{v \in V \setminus W \,|\, \exists u \in W \,|\, (u, v) \in E\}$. Именно к поиску таких подмножеств сводятся реализации расширения и пересечения сеточных подобластей в соответствии с (3), (4) и (9).

Набор вопросов, с которыми сталкивается разработчик параллельного программного обеспечения МДО при алгебраическом способе постановки задачи, включает в себя (но не ограничивается только ими!) следующие задачи:

— декомпозиция матрицы $A$ на подматрицы, т.е. выделение подграфов в матричном графе, причем как можно более одинаковых по размерности (количеству вершин) для последующей равномерной загрузки MPI-процессов (на языке геометрического подхода эта операция есть не что иное как декомпозиция исходной области на подобласти без пересечения (3) или с пересечениями (9)); некоторая алгоритмическая информация содержится в [9];

— построение (выделение) смежных подмножеств в подграфах; определение смежных подмножеств графа фактически используется при построении сеточных границ и расширенной подобласти $\Omega_q^{\Delta}$, при этом $\mathrm{Adj}(W)$ соответствует границе $\Gamma_q^1$ из (9), если в качестве $W$ взять саму подобласть $\Omega_q$;

— нахождение пересечения двух подграфов (двух дискретных множеств, если подграф представлен совокупностью своих вершин); эта операция требуется для осуществления обменов информацией между подобластями;

— проведение операций сужения–продолжения (пролонгации) на подграфах;

— организация эффективного хранения информации о подграфах (подобластях) и обменной информации для коммуникации между ними в процессе итерационного решения исходной задачи.

Нельзя не упомянуть в контексте графового представления матриц, что по этой тематике существуют достаточно известные библиотеки, которые решают указанный круг вопросов, например, ParMetis [10] — реализованная с помощью MPI параллельная версия библиотеки METIS алгоритмов над графами [11].

Каждая из перечисленных выше задач является нетривиальной с точки зрения эффективной параллельной программной реализации, что требует, в свою очередь, самостоятельного детального описания и выходит за рамки тематики статьи. Поэтому отметим только некоторые принципиальные аспекты этих задач.

Так, относительно трудоемкости нахождения пересечения двух подграфов известно, что если массивы, перечисляющие вершины подмножеств $U$, $W$ некоторого графа $V$, предварительно не отсортированы, то алгоритм пересечения требует $O\big(|U|+|W|\big)$ действий и $O\big(|V|\big)$ дополнительной памяти, инициализация которой, кстати, имеет сложность также $O\big(|V|\big)$ [9]. В случае же, когда проведена сортировка массивов, можно применять как минимум две реализации, одна из которых имеет сложность $O\big(|U|+|W|\big)$, а вторая — $O\Big(\min\big(|U|,|V|\big)\log\Big(\max\big(|U|,|V|\big)\Big)\Big)$. Обе требуют лишь $O(1)$ дополнительной памяти. Именно в этой связи производятся локальные переупорядочивания узлов в сеточных подобластях и их взаимных пересечениях.

Для эффективности параллельной реализации МДО большое значение имеет схема хранения (распределения) векторных и матричных блоков по MPI-процессам. Введем следующие обозначения:

$$\Omega_q^{\Delta} = \Omega_q \bigcup_{r\in\widehat{\omega}_q} \Omega_{q,r}, \quad \Omega_{q,r} = \Omega_q^{\Delta} \bigcap \Omega_r, \tag{17}$$

где $\Omega_q$ соответствует декомпозиции (3) без пересечения подобластей, а $\Omega_{q,r}$ — пересечения расширенной подобласти с ее “нерасширенными” смежными подобластями. В общем случае декомпозиции с пересечением подобластей, в соответствии с описанной выше в разделе 3 структурой данных, представляется целесообразным следующее размещение данных: каждый $q$-й MPI-процесс хранит те компоненты искомого и заданного векторов $u$, $f$, которые относятся к множеству $\Omega_q$, а также (по отдельности) из смежных подмножеств $\Omega_{q,r} = \Omega_q^{\Delta} \bigcap \Omega_r$ в соответствии с обозначениями в (17). Аналогично размещаются диагональные и внедиагональные блоки $A_{q,q}$ и $A_{q,r}$ матричной блочной строки $A_q$, причем пересылок их значений во время итераций МДО не требуется.

**5. Заключение.** В настоящей статье рассмотрены разнообразные аспекты такой комплексной проблемы, как создание вычислительно-информационных методов и технологий, реализуемых в составе высокопроизводительного математического и программного обеспечения для решения широкого класса больших разреженных СЛАУ, возникающих при аппроксимациях многомерных краевых задач на неструктурированных сетках, на кластерных МВС с помощью параллельных МДО. Предложены некоторые подходы к эффективному отображению двухуровневых итерационных алгоритмов на компьютерную архитектуру с иерархической памятью, частично реализованные, например, в составе библиотеки KRYLOV [13]. Широкий класс методов декомпозиции областей представлен как многоликая сеточно-геометрическая, матрично-алгебраическая и информационно-графовая структуры, взаимно обогащающие данное актуальное направление исследований.

Работа поддержана грантом Российского научного фонда № 14–11–00485, информационно-технологическая часть поддержана грантом РФФИ № 14–07–00128.

СПИСОК ЛИТЕРАТУРЫ

1. *Rabenseifner R., Hager G., Jost G.* Hybrid MPI and OpenMP parallel programming. MPI + OpenMP and other models on clusters of SMP nodes. Tutorial tut123 at SC13, November 17, 2013, Denver (CO) USA. http://openmp.org/sc13/HybridPP_Slides.pdf
2. *Ильин В.П.* DELAUNAY: технологическая среда генерации сеток // Сибирский журн. индустриальной математики. 2013. **16**, № 2. 83–97.
3. Domain Decomposition Methods. http://ddm.org.
4. *Toselli A., Widlund O.B.* Domain decomposition methods: algorithms and theory. Heidelberg: Springer, 2005.
5. *Gurieva Y.L., Il'in V.P.* On parallel computational technologies of augmented domain decomposition methods // Lecture Notes in Computer Science. Vol. 9251. Heidelberg: Springer, 2015. 35–46.
6. *Dolean V., Jolivet P., Nataf F.* An introduction to domain decomposition methods: algorithms, theory, and parallel implementation. https://hal.archives-ouvertes.fr/cel-01100932v4/document.
7. *Ильин В.П.* Параллельные методы и технологии декомпозиции областей // Вестник ЮУрГУ. Серия "Вычислительная математика и информатика". 2012. № 46. 31–44.
8. *Климонов И.А., Корнеев В.Д., Свешников В.М.* Технологии распараллеливания решения трехмерных краевых задач на квазиструктурированных сетках в гибридной вычислительной среде CPU+GPU // Вычислительные методы и программирование: Новые вычислительные технологии. 2016. **17**. 65–71.
9. *Ильин В.П., Перевозкин Д.В.* О некоторых вариантах метода декомпозиции областей // Вестник ЮУрГУ. Серия "Вычислительная математика и информатика". 2014. **3**, № 2. 5–19.
10. http://glaros.dtc.umn.edu/gkhome/metis/parmetis/overview.
11. *Karypis G.* METIS — serial graph partitioning and fill-reducing matrix ordering. http://glaros.dtc.umn.edu/gkhome/metis/metis/overview.
12. *Писсанецки С.* Технология разреженных матриц. М.: Мир, 1988.
13. *Бутюгин Д.С., Гурьева Я.Л., Ильин В.П., Перевозкин Д.В., Петухов А.В., Скопин И.Н.* Функциональность и технологии алгебраических решателей в библиотеке Krylov // Вестник ЮУрГУ. Серия "Вычислительная математика и информатика". 2013. **2**, № 3. 92–105.

Поступила в редакцию
25.03.2016

---

## Algebraic-Geometric and Information Structures of Domain Decomposition Methods

**Y. L. Gurieva [1], V. P. Il'in [2], and D. V. Perevozkin [3]**

[1] *Institute of Computational Mathematics and Mathematical Geophysics, Siberian Branch of Russian Academy of Sciences; prospekt Lavrentyeva 6, Novosibirsk, 630090, Russia; Ph.D., Senior Scientist, e-mail: yana@lapasrv.sscc.ru*

[2] *Institute of Computational Mathematics and Mathematical Geophysics, Siberian Branch of Russian Academy of Sciences; prospekt Lavrentyeva 6, Novosibirsk, 630090, Russia; Dr. Sci., Professor, Principal Scientist, e-mail: ilin@sscc.ru*

[3] *Institute of Computational Mathematics and Mathematical Geophysics, Siberian Branch of Russian Academy of Sciences; prospekt Lavrentyeva 6, Novosibirsk, 630090, Russia; Junior Scientist, e-mail: dperevozkin@mail.ru*

Received March 25, 2016

**Abstract:** Algebraic, geometric, and informational aspects of parallel decomposition methods are considered to solve large systems of linear equations with sparse matrices which arise under approximation of multidimensional boundary value problems on unstructured grids. Algorithms are based on a partition of a grid computational domain into its subdomains with the parameterized value of overlapping and various interface conditions on the adjacent boundaries. Some questions arising in algebraic decomposition of the original matrix are discussed. Various two-level iterative processes are used. They include both preconditioned Krylov methods with a coarse grid correction

and parallel solution of auxiliary subsystems in subdomains by direct or iterative algorithms. Parallelization of algorithms is implemented by means of a hybrid programming with separate MPI processes for every subdomain and by multithreaded computations over shared memory in each of the subdomains. Communications between adjacent subdomains are carried out on each external iteration via preliminary creation of some exchange buffers and a usage of non-blocking operations which make possible to combine both the arithmetic operations and data transfer.

**Keywords:** domain decomposition, large linear systems, sparse matrices, data structures, hybrid programming, parallel programming.

## References

1. R. Rabenseifner, G. Hager, and G. Jost, "Hybrid MPI and OpenMP Parallel Programming. MPI+OpenMP and Other Models on Clusters of SMP nodes," Tutorial tut123 at SC13, November 17, 2013, Denver (CO) USA. http://openmp.org/sc13/HybridPP_Slides.pdf. Cited April 5, 2016.

2. V. P. Il'in, "DELAUNAY: A Technological Environment for Grid Generation," Sib. Zh. Ind. Mat. **16** (2), 83–97 (2013).

3. Domain Decomposition Methods. http://www.ddm.org. Cited April 5, 2016.

4. A. Toselli and O. B. Widlund, *Domain Decomposition Methods: Algorithms and Theory* (Springer, Heidelberg, 2005).

5. Y. L. Gurieva and V. P. Il'in, "On Parallel Computational Technologies of Augmented Domain Decomposition Methods," in *Lecture Notes in Computer Science* (Springer, Heidelberg, 2015), Vol. 9251, pp. 35–46.

6. V. Dolean, P. Jolivet, and F. Nataf, "An Introduction to Domain Decomposition Methods: Algorithms, Theory, and Parallel Implementation,"
https://hal.archives-ouvertes.fr/cel-01100932v4/document. Cited April 5, 2016.

7. V. P. Il'in, "Parallel Methods and Technologies of Domain Decomposition," Vestn. South Ural Univ. Ser. Vychisl. Mat. Inf. No. 46, 31–44 (2012).

8. I. A. Klimonov, V. D. Korneev, and V. M. Sveshnikov, "Parallelization Technologies for Solving Three-Dimensional Boundary Value Problems on Quasi-Structured Grids Using the CPU+GPU Hybrid Computing Environment," Vychisl. Metody Programm. **17**, 65–71 (2016).

9. V. P. Il'in and D. V. Perevozkin, "On Some Variants of Domain Decomposition Methods," Vestn. South Ural Univ. Ser. Vychisl. Mat. Inf. **3** (2), 5–19 (2014).

10. ParMETIS — Parallel Graph Partitioning and Fill-Reducing Matrix Ordering.
http://glaros.dtc.umn.edu/gkhome/metis/parmetis/overview. Cited April 5, 2016.

11. G. Karypis, "METIS — Serial Graph Partitioning and Fill-Reducing Matrix Ordering,"
http://glaros.dtc.umn.edu/gkhome/metis/metis/overview. Cited April 5, 2016.

12. S. Pissanetzky, *Sparse Matrix Technology* (Academic, London, 1984; Mir, Moscow, 1988).

13. D. S. Butyugin, Y. L. Guryeva, V. P. Il'in, et al., "Parallel Algebraic Solvers Library Krylov," Vestn. South Ural Univ. Ser. Vychisl. Mat. Inf. **2** (3), 92–105 (2013).