УДК 519.6

О ВОПРОСАХ РАСПАРАЛЛЕЛИВАНИЯ КРЫЛОВСКИХ ИТЕРАЦИОННЫХ МЕТОДОВ[1]

В.П. Ильин

В работе рассматриваются математические вопросы многообразных вычислительных технологий методов распараллеливания итерационных процессов крыловского типа для решения больших разреженных симметричных и несимметричных СЛАУ, возникающих при сеточных аппроксимациях многомерных краевых задач для систем дифференциальных уравнений. Характерным примером являются конечно-элементные приближения в газогидродинамических приложениях, где в каждом узле определены пять неизвестных функций, в силу чего СЛАУ имеет мелкоблочную структуру. Основой применяемых алгоритмов является гибкий метод обобщенных минимальных невязок FGMRES с динамическими предобуславливателями аддитивного типа, представляющий собой верхний уровень двухступенчатого итерационного алгоритма Шварца.

Для повышения производительности алгебраических решателей автором предлагается применение различных подходов: декомпозиции расчетной области с различными топологиями, типами краевых условий на смежных границах и размерами пересечений подобластей, методов грубосеточной коррекции и агрегации, дефляции и неполной факторизации матриц. Описываются унифицированные формулировки используемых алгоритмов, а также вопросы их вычислительной эффективности и масштабируемого распараллеливания на суперкомпьютерах гетерогенной архитектуры. Приводятся примеры технологических требований к особенностям программных реализаций библиотек параллельных алгоритмов для решения систем линейных алгебраических уравнений.

Ключевые слова: итерационные методы, подпространства Крылова, предобусловленные матрицы, декомпозиция областей, параллельные алгоритмы, программные и вычислительные технологии.

Введение

В последние годы сформировалось бурно развивающееся направление вычислительной алгебры, связанное с решением больших разреженных систем линейных алгебраических уравнений (СЛАУ), возникающих из сеточных аппроксимаций (конечно-элементных или конечно-объемных, см. [1] и цитируемые там работы) многомерных краевых задач для систем дифференциальных уравнений в частных производных. Естественно, это связано с актуальными проблемами математического моделирования для сложных междисциплинарных приложений, включающих гидро-газодинамические процессы, динамику напряженно-деформированных состояний и др., на современных архитектурах суперкомпьютеров петафлопных масштабов. Само понятие большой задачи быстро эволюционирует, и сейчас необходимо говорить о решениях СЛАУ с порядками 10^{10} на многопроцессорных вычислительных системах (МВС) с числом ядер, или потоков, в десятки и сотни тысяч.

Повышение производительности вычислений, или быстродействия, в рассматриваемой прикладной сфере обуславливается двумя основными факторами. Первый — это конструирование итерационных процессов с высокой скоростью сходимости (прямые алгоритмы применимы только для относительно небольших размерностей задач), а второй — технологическое обеспечение масштабируемости распараллеливания, что означает сохранение

[1]Статья рекомендована к публикации программным комитетом Международной научной конференции «Параллельные вычислительные технологии 2013».

эффективности с ростом порядков СЛАУ и/или количества вычислительных устройств. Архитектура последних развивается главным образом количественно, а не качественно, и типовая структура гетерогенной МВС — это кластер с одинаковыми или почти одинаковыми вычислительными узлами, каждый из которых содержит несколько центральных процессорных многоядерных элементов (CPU) с общей памятью, а также несколько специализированных ускорителей, среди которых наибольшее распространение получили «графические процессорные элементы общего назначения» (GPGPU, или просто GPU) с очень большим числом ядер и сложной организацией внутренней иерархической оперативной памяти. Реализация параллельных вычислений на такого типа МВС осуществляется средствами систем MPI, OpenMP и CUDA, причем оптимизация программного кода фактически производится разработчиком в основном вручную, поскольку соответствующие средства автоматизации недостаточно функциональны.

Главными инструментами для решения возникающих здесь алгоритмических проблем являются методы декомпозиции областей [2–8] и итерационные процессы в подпространствах Крылова [1, 5]. Хотя эти направления уже стали классическими в вычислительной математике, они сейчас переживают период бурного развития, и буквально каждый год появляются новые интересные идеи. Рост эффективности алгебраических решателей достигается с помощью применения многообразных подходов: оптимизация размеров и топологий пересечений соседних подобластей, а также типов краевых условий на смежных границах, методы грубосеточной коррекции и агрегации, дефляции и неполной факторизации.

В разделе 1 мы даем постановку и особенности рассматриваемых задач, а также сравнительный анализ методов их решения. Раздел 2 посвящен как общим, так и специальным технологическим аспектам распараллеливания алгоритмов, а в последнем разделе описываются примеры технологических требований к программным реализациям алгебраических решателей и, в частности, к библиотеке KRYLOV, предварительные сообщения о которой опубликованы в [6, 8]. В заключении отмечается комплексность взаимосвязанных теоретических, алгоритмических и технологических вопросов, определяющих уровень высокопроизводительных вычислений математического и программного обеспечения для решения актуальных алгебраических систем.

1. Математические аспекты двухуровневых методов декомпозиции

Рассматривается следующая задача: пусть вещественная матрица СЛАУ

$$Au = f, \;\; A = \{a_{i,j}\} \in \mathcal{R}^{N,N}, \tag{1}$$

разбита на блочные строки $A = \{A_p \in \mathcal{R}^{N_p,N}, \;\; p = 1, ..., P\}, \;\; N_1 + ... + N_p = N$, с приблизительно равным числом строк $N_p \gg 1$ в каждой. Соответствующим образом также разбиваются векторы искомого решения и правой части $u = \{u_p\}, \;\; f = \{f_p\}$, так что система уравнений (1) может быть записана в форме совокупности P подсистем

$$A_{p,p}u_p + \sum_{\substack{q=1 \\ q \neq p}}^{P} A_{p,q}u_q = f_p, \;\; p = 1, ..., P, \tag{2}$$

где $A_{p,q} \in \mathcal{R}^{N_p,N_q}$ — прямоугольные матричные блоки, получаемые при разбиении каждой блочной строки (и всей матрицы A) на блочные столбцы.

Пусть матрица (1) задана в сжатом разреженном CSR-формате [3] с указанием числа строк N_p в каждой из P подсистем (2), в соответствии с разбиением матрицы A на блочные строки A_p. Естественно предполагается, что строки исходной матрицы пронумерованы подряд от 1 до N, так что номера строк каждого блока A_p меняются от $1+\sum\limits_{i=1}^{p-1} N_i$ до $\sum\limits_{i=1}^{p} N_i$ (эти номера назовем глобальными).

Предполагается, что СЛАУ (1) представляет собой систему сеточных уравнений, так что каждая компонента векторов u, f соответствует узлу сетки, общее число которых в расчетной сеточной области $\Omega^h = \bigcup\limits_{p=1}^{P} \Omega_p$ равно N. При этом блочное разбиение матриц и векторов соответствует разбиению (декомпозиции) Ω^h на P сеточных непересекающихся подобластей Ω_p, в каждой из которых находится N_p узлов.

Описанная декомпозиция Ω^h не использует узлов — разделителей, т.е. условные границы подобластей не проходят через узлы сетки. Формальности ради можно считать, что внешность расчетной области есть неограниченная подобласть Ω_0 с числом узлов $N_0 = 0$.

Сеточное уравнение для i-го узла сетки может быть записано в виде

$$a_{i,i}u_i + \sum_{\substack{j\in\omega_i \\ j\neq i}} a_{i,j}u_j = f_i, \;\; i \in \Omega^h, \tag{3}$$

где через ω_i обозначается сеточный шаблон i-го узла, т.е. совокупность номеров всех узлов, участвующих в i-м уравнении.

Будем считать, что сеточные узлы и соответствующие переменные пронумерованы следующим образом: сначала идут подряд все узлы 1-й подобласти Ω_1 (неважно в каком внутреннем порядке), затем — узлы второй подобласти и т.д. Отметим, что взаимнооднозначного соответствия алгебраической и геометрической (сеточной) интерпретации структуры СЛАУ может и не существовать. Типичный пример — одному узлу сетки может соответствовать $m > 1$ переменных в алгебраической системе. Если для каждого узла такие кратные переменные нумеруются подряд, то структура СЛАУ приобретает мелкоблочный ($m \ll N$) вид (в (3) u_i и f_i означают не числа, а подвекторы порядка m соответственно, $a_{i,j} \in \mathcal{R}^{m,m}$), и на таких случаях мы остановимся в последующем особо.

1.1. О построении параллельных предобуславливателей

Для заданного блочного разбиения СЛАУ, или декомпозиции сеточной расчетной области без пересечения подобластей, можно построить хорошо распараллеливаемый аддитивный метод Шварца, представимый итерационным процессом «по подобластям»:

$$A_{p,p}u_p^n = f_p - \sum_{\substack{q=1 \\ q\neq p}}^{P} A_{p,q}u_q^{n-1} \equiv g_p^{n-1}. \tag{4}$$

Здесь n будем называть номером внешней итерации, которая в действительности реализуется более утонченным образом, чему будет посвящен специальный раздел.

В целом алгоритм решения крупноблочной СЛАУ является двухуровневым и нижний, или внутренний, уровень заключается в решении (прямым или итерационным методом) для каждого n — независимо и параллельно — подсистем вида (4) с матрицами $A_{p,p}$.

При вычисленных правых частях g_p^{n-1} нахождение всех u_p^n требует для каждой подобласти решения подсистем с порядками N_p, которые на каждой n-й итерации могут выполнять-

ся независимо и одновременно на разных процессорах. В данном случае перевычисление g_p^{n-1} фактически означает использование новых значений u_p^{n-1} для соседних подобластей в качестве граничного условия 1-го рода (Дирихле) для околограничных внутренних узлов из Ω_p.

Известно, что скорость сходимости итераций метода Шварца (4) возрастет, если декомпозицию расчетной области сделать с пересечением подобластей.

В силу этого мы дадим определение расширенной сеточной подобласти $\bar{\Omega}_p \supset \Omega_p$, имеющей пересечения с соседними подобластями, величину которых мы будем определять в терминах количества околограничных сеточных слоев, или фронтов. Обозначим через $\Gamma_p^0 \in \Omega_p$ множество внутренних околограничных узлов из Ω_p, т.е. таких узлов $P_i \in \Omega_p$, у которых один из соседей не лежит в $\Omega_p (P_j \notin \Omega_p,\ j \in \omega_i,\ j \neq i)$. Обозначим далее через Γ_p^1 множество узлов, соседних с узлами из Γ_p^0, но не принадлежащих Ω_p, через Γ_p^2 – множество узлов, соседних с узлами из Γ_p^1, но не принадлежащих объединению $\Gamma_p^1 \bigcup \Omega_p$, через Γ_p^3 – множество узлов, соседних с Γ_p^2, но не принадлежащих $\Gamma_p^2 \bigcup \Gamma_p^1 \bigcup \Omega_p$, и т.д. Соответственно эти множества назовем первым внешним слоем (фронтом) узлов, вторым слоем, третьим и т.д. Получаемое объединение узлов

$$\bar{\Omega}_p \equiv \Omega_p \bigcup \Gamma_p^1 ... \bigcup \Gamma_p^{\Delta}$$

будем называть расширенной p-й сеточной подобластью, а целую величину Δ определяем как величину расширения, или пересечения (в терминах количества сеточных слоев). Случай $\Delta = 0$ фактически означает декомпозицию области Ω^h на подобласти без пересечений $(\Omega_p^0 = \Omega_p)$.

На формальном алгебраическом языке каждой расширенной подобласти можно сопоставить подсистему уравнений

$$(\bar{A}_{p,p} + \theta \bar{D}_p)\bar{u}_p = \bar{f}_p - \sum_{\substack{q=1 \\ q \neq p}}^{P} \bar{A}_{p,q}\bar{u}_q + \theta \bar{D}_p \bar{u}_p \tag{5}$$

где $\bar{D}_p$ — диагональная матрица, определяемая соотношением

$$\bar{D}_p e = \sum_{\substack{q=1 \\ q \neq p}}^{P} \bar{A}_{p,q} e,\ \ e = (1, ..., 1)^T \in \mathcal{R}^{\bar{N}_p}.$$

Здесь размерность векторов $\bar{u}_p^n$ и $\bar{f}_p$ равна числу узлов $\bar{N}_p$ в расширенной подобласти $\bar{A}_{p,p}, \bar{D}_p \in \mathcal{R}^{\bar{N}_p, \bar{N}_p}$, а введенный параметр θ формально позволяет рассматривать зависящие от $\bar{u}_p$ и $\bar{u}_q$ правые части (5) как аппроксимации краевых условий на смежных границах подобластей: $\theta = 0$ соответствует условию 1-го рода (Дирихле), $\theta = 1$ — условию 2-го рода (Неймана), а $\theta \in (0, 1)$ — условию 3-го рода (Робена) [10].

Переходя теперь к полному вектору u и вводя блочно-диагональные матрицы

$$B_p = block\text{-}diag\{\bar{A}_{p,p} + \theta \bar{D}_p\}, \tag{6}$$

из (5) получаем итерационный процесс вида

$$\begin{aligned} &u^n = u^{n-1} + B^{-1}(f - Au^{n-1}) = u^{n-1} + B^{-1}r^{n-1} = \\ &= Tu^{n-1} + \bar{f},\ \ T = I - B^{-1}A,\ \ \bar{f} = B^{-1}f, \end{aligned} \tag{7}$$

где B — предобуславливающая матрица, определяемая следующим образом.

Введем матрицу $W_p = (w_1, ..., w_{\bar{N}_p})^T \in \mathcal{R}^{\bar{N}_p, N}$ «сужения» вектора $u \in \mathcal{R}^N$ в пространство $\mathcal{R}^{\bar{N}_p}$ соответствующей подобласти, для чего компоненты каждой вектор-строки $w_k = \{w_{k,i}\}$ полагаются равными единице, если $i \in \bar{\Omega}_p$, и нулю в противном случае. При этом W_p^T является матрицей оператора продолжения пространства $\bar{\Omega}_p$ в Ω, и в итоге предобуславливатель аддитивного метода Шварца (additive Swartz) принимает вид

$$B^{-1} = B_{AS}^{-1} = \sum_{p=1}^{P} \bar{B}_p^{-1}, \;\; \bar{B}_p^{-1} = W_p^T B_p^{-1} W_p. \tag{8}$$

На алгебраическом языке формулы (7) при $B = B_{AS}$ определяют стационарный блочный метод Якоби (BJ), на каждом шаге которого надо обращать матрицу B_p, что фактически означает одновременное (параллельное) решение расширенных СЛАУ в подобластях. Если эти процедуры осуществляются с помощью «внутренних» итерационных процессов, то это неизбежно приводит к тому, что в (7) на каждом n-м шаге реально используется переменное (динамическое) предобуславливание с матрицами B_n.

Кардинальный подход к ускорению итераций заключается в переходе от формул (7) для BJ к алгоритмам в предобусловленных подпространствах Крылова $\mathcal{K}_n(v^0, v^1, ..., v^{n-1})$, где введены обозначения $v^0 = B_0^{-1} r^0, v^k = B_k^{-1} A v^{k-1}$. Один из возможных путей здесь заключается в применении гибкого (flexible) метода обобщенных минимальных невязок FGMRES [5], который на каждом n-м шаге минимизирует норму $||r^n||$, но требует для своей реализации оперативную память объемом $2nN$. Использование FGMRES с рестартами через каждые m итераций при больших n позволяет значительно сокращать объем требуемой памяти до $2mN$, но существенно замедляет скорость сходимости.

Главный, и практически единственный, путь повышения эффективности алгоритмов — это развитие крыловских методов или за счет совершенствования предобуславливателей, или путем улучшения итерационных подпространств (добавление новых базисных векторов, в частности, что тоже может интерпретироваться как введение дополнительного предобуславливания).

Одним из резервов ускорения является грубосеточная коррекция (coarse grid correction, CGC), или агрегирование, а также идейно примыкающие сюда алгебраические многосеточные методы (algebraic multi-grid, AMG). Суть здесь заключается в том, что на каждом шаге стандартного BJ эволюция последовательных приближений от каждой области передается только ее непосредственным соседям. Идея исправления данной ситуации — формирование и решение дополнительных относительно небольших СЛАУ, которые соединяли бы все подобласти и передавали бы, пусть грубо, глобальные итерационные возмущения.

Реализация данного подхода заключается в конструировании дополнительного предобуславливателя и внешне аналогична (8). Мы формируем новый оператор сужения с матрицей $W_c \in \mathcal{R}^{N_c, N}$, $N_c \ll N$, в которой каждая строка содержит только по несколько ненулевых элементов (обычно равных единице), соответствующих одной из подобластей. Транспонированная матрица W_c^T будет представлять тогда оператор продолжения, грубосеточный предобуславливатель принимает вид

$$B_c^{-1} = W_c^T A_c^{-1} W_c, \;\; A_c = W_c A W_c^T \in \mathcal{R}^{N_c, N_c}, \tag{9}$$

и полный предобуславливатель определяется аддитивным образом:

$$B^{-1} = B_{AS}^{-1} + B_c^{-1}. \tag{10}$$

Отметим, что реализация CGC на каждой n-й итерации требует дополнительного решения СЛАУ с матрицей невысокого порядка A_c, что можно сделать с помощью прямого решателя, например, PARDISO из библиотеки MKL INTEL [9].

Развитие идей грубосеточной коррекции активно продолжается в различных направлениях, см. [11],bib12 и цитируемые там работы. Многосеточные методы используют технологии интерполяции на каждом из последовательных этапов дискретизации. Адаптивные алгоритмы сглаженного агрегирования основаны на взвешенных усреднениях различных приближений. Методы типа FETI применяют декомпозицию области с явным выделением разделителей сеточных подмножеств (макрограни, макроребра, вершины) и построение дополнений Шура на иерархическом принципе.

Опишем еще один подход к ускорению крыловских процессов — метод дефляции, который имеет широкое распространение в разных версиях. Мы его представим в применении к выбору начального приближения, если СЛАУ с одинаковой матрицей решается многократно с разными правыми частями. Именно такая ситуация возникает в двухуровневых методах декомпозиции областей. Пусть в результате предыдущего решения СЛАУ с помощью какого-то крыловского метода вычислены A-ортогональные векторы $(w_1, ..., w_m) = W_d$, составляющие базис пространства, которое будем называть дефляционным. Для решения новой системы выбираем начальное приближение u^0 таким, чтобы соответствующая начальная невязка r^0 и начальный направляющий вектор p^0 удовлетворяли условиям ортогональности

$$W_d^T r^0 = 0, \;\; W_d^T A p^0 = 0. \tag{11}$$

Допустим, что u^{-1} есть «предварительный» начальный вектор, а $r^{-1} = f - Au^{-1}$ — соответствующая невязка. Тогда условия (11) будут выполняться, если положить

$$\begin{array}{l} u^0 = u^{-1} + W_d A_d^{-1} W_d^T r^{-1}, \;\; r^0 = f - Au^0, \\ p^0 = [I - W_d A_d^{-1}(AW_d^T)]r^0, \;\; A_d = W_d^T A W_d. \end{array} \tag{12}$$

В рамках одной статьи невозможно дать содержательный обзор современных тенденций в развитии предобусловленных итерационных процессов крыловского типа. Можно только констатировать, что оригинальные подходы появляются практически непрерывно, и данный момент следует рассматривать как важный технологический фактор при создании математического и программного обеспечения, ориентированного на длительный жизненный цикл.

В заключение данного раздела отметим, что если в матрице исходной СЛАУ (1) выделить главную блочную диагональ ($A = D - C, \;\; D = \{A_{p,p}\}$), то при условии ее разложения на блочно-треугольные множители систему можно переписать в следующем виде:

$$\begin{array}{l} Du = Cu + f, \;\; D = L_D U_D = (L_D L_D^t), \\ U_D u = L_D^{-1} C U_D^{-1} U_D u + L_D^{-1} f. \end{array} \tag{13}$$

Отсюда можно ввести формально предобусловленную СЛАУ

$$\begin{array}{l} \bar{A}\bar{u} \equiv (I - \bar{T})\bar{u} = \bar{f}, \;\; u = U_D^{-1}\bar{u}, \\ \bar{T} = L_D^{-1} C U_D^{-1} (= \bar{T}^t), \;\; \bar{f} = L_D^{-1} f, \end{array} \tag{14}$$

для решения которой естественно применять блочный (двусторонне предобусловленный) метод Якоби

$$\bar{u}^n = \bar{T}\bar{u}^{n-1} + \bar{f}, \;\; \bar{u}^n = U_D u^n. \tag{15}$$

Заметим, что если матрица A симметрична, то этим же свойством обладают и матрицы $T, \bar{A}$. Отметим, что в формулах (13) – (15) можно положить

$$L_D = D, \;\; U_D = I, \text{ или } \;\; L_D = I, \;\; U_D = D,$$

что будет соответствовать одностороннему предобуславливанию (левому или правому соответственно).

1.2. Некоторые особенности методов в подпространствах Крылова

Как видно из предыдущего раздела, итерационные методы (мы подразумеваем «в подпространствах Крылова» как оптимальные) можно применять для решения как «глобальной» СЛАУ, т.е. для ускорения блочного метода Якоби, так и для «локальных» систем в подобластях. В дальнейшем крыловские алгоритмы мы рассмотрим в единообразной форме, независимо от того, применяются они на внешних или внутренних итерациях, основой которой является A^s-ортогонализация различных векторов, $s = 0$ или $s = 1$, см. [**?**].

Если предобусловленная СЛАУ $\bar{A}\bar{u} = \bar{f}$ симметрична, то наиболее экономичными являются методы сопряженных градиентов или сопряженных невязок (CG или CR, для $s = 0$ или $s = 1$ соответственно), описываемые следующими формулами:

$$\begin{array}{l} r^0 = \bar{f} - \bar{A}\bar{u}^0 = \hat{u}^1 - \bar{u}^0, \;\; \hat{u}^1 = T\bar{u}^0 + \bar{f}, \;\; p^0 = r^0, \\ \bar{u}^{n+1} = \bar{u}^n + \alpha_n^{(s)} p^n, \;\; \alpha_n^{(s)} = \rho_n^{(s)}/\delta_n^{(s)}, \;\; \rho_n^{(s)} = (\bar{A}^s r^n, r^n), \\ \delta_n^{(s)} = (\bar{A}p^n, \bar{A}^{(s)} p^n), \;\; r^{n+1} = r^n - \alpha_n^{(s)} \bar{A} p^n, \\ p^{n+1} = r^{n+1} + \beta_n^{(s)} p^n, \;\; \beta_n^{(s)} = \rho_{n+1}^{(s)}/\rho_n^{(s)}. \end{array} \tag{16}$$

Различные методы решения несимметричных систем базируются на A^s-ортогонализации Арнольди с предобуславливанием. Поскольку в общем случае предобуславливающие матрицы B_n могут отличаться на разных итерациях, то мы рассматриваем «гибкую» (flexible) ортогонализацию в следующем виде:

$$\begin{array}{l} u^n = u^0 + y_1 v^1 + ... + y_n v^n, \;\; (v^n, A^s v^k) = d_n^{(s)} \delta_{k,n}, \\ d_n^{(s)} = (v^n, A^s v^n), \\ v^{n+1} = AB_n^{-1} v^n - \sum_{k=1}^n h_{k,n}^{(s)} v^k, \;\; v^1 = r^0 = f - Au^0, \;\; n = 1, 2, ... \\ h_{k,n}^{(s)} = \frac{(Av^n, A^s v^k)}{(A^s v^k, v^k)}, \;\; k = 1, ..., n+1, \;\; V_{n+1} = (v^1, ..., v^{n+1}) \\ \bar{H}_n = \{h_{k,n}\} = \left[\begin{array}{c} H_n \\ e_n^t \end{array} \right] \in \mathcal{R}^{n+1,n}, \;\; H_n \in \mathcal{R}^{n,n}. \end{array} \tag{17}$$

Получаемые предобусловленные векторы $B_k^{-1} v_k$ образуют базис подпространства Крылова

$$\mathcal{K}_{n+1}(r^0, A) = \text{span}\{B_1^{-1} v^1, ..., B_{n+1}^{-1} v^{n+1}\} = \text{span}\{B_1^{-1} r^0, AB_2^{-1} r^0, ..., A^n B_{n+1}^{-1} r^0\}, \tag{18}$$

и на их основе формируются алгоритмы полной A^s-ортогонализации или методы обобщенных A^s-минимальных невязок (FOM или A-FOM, GMRES или A-GMRES), см. [1, 5].

Внешние итерации оканчиваются по выполнению условия

$$(r^n, r^n) \le \varepsilon_{ex}^2 (\bar{f}, \bar{f}),$$

где $\varepsilon_{ex} \ll 1$ характеризует точность решения. Оптимизация критериев останова внутренних итераций — это непростой вопрос, и здесь параметры точности ε_{in}^n в принципе должны зависеть от номера n внешней итерации.

2. Технологические вопросы реализации параллельных итерационных алгоритмов

Рассматриваемые в данном разделе вопросы — это конкретизация общей проблемы «отображение алгоритмов на архитектуру МВС». В качестве типовой модели вычислительной системы может служить НКС-30Т — гетерогенный кластер ИВМиМГ СО РАН [13]. Такая МВС формально представляет набор вычислительных узлов, которые с помощью коммуникационной сети обеспечиваются связями «каждый с каждым», управляемыми с помощью программных средств системы передачи сообщений MPI [14]. Каждый узел имеет свою многоуровневую память (большую оперативную и поменьше — сверхбыстрый кэш, тоже имеющий внутренние уровни), общую для расположенных в нем многоядерных центральных процессорных устройств (CPU, в НКС-30Т их два, каждый с четырьмя или шестью ядрами), а также графические процессорные устройства с очень большим числом ядер (GPU, в НКС-30Т их три, с 512 ядрами в каждом). На каждом CPU (и даже на их отдельных ядрах) можно формировать вычислительный MPI-процесс, внутри которого распараллеливание реализуется средствами системы OpenMP [15] над общей памятью (более конкретно — организуется несколько вычислительных потоков). На одном GPU допускается запуск только одного MPI-процесса, причем внутри него программирование осуществляется на языке CUDA [16] с достаточно сложными средствами управления внутренней иерархической памятью. Особенностью (и недостатком) GPU является относительно медленные коммуникации с памятью CPU. В целом средства MPI обеспечивают организацию синхронных или асинхронных вычислений, в том числе при совмещении их во времени с передачей данных от процессора к процессору.

Как видно из приведенного описания, построение даже грубой математической модели вычислительного процесса на такой архитектуре, с целью его оптимизации и оценок коммуникационных потерь, не представляется возможным. Поэтому соответствующие исследования являются сугубо экспериментальными, а основной инструмент в данном случае — это метод проб и ошибок.

Справедливости ради следует сказать, что математик-программист работает не с пустыми руками, а при наличии достаточного богатого вычислительного инструментария (главным образом библиотеки BLAS и SPARSE BLAS [8], содержащие основные алгебраические операции с векторами и матрицами, в том числе разреженными), созданного профессионалами с помощью экономичных языковых средств низшего уровня. В частности, можно упомянуть библиотеку CUSP [17] для решения на GPU разреженных СЛАУ с использованием средств BLAS.

По поводу вопросов реализации параллельных итерационных алгоритмов мы отметим две проблемы, относящиеся к области вычислительных технологий. Первая из них касается формирования вспомогательных СЛАУ для подобластей, которые необходимо решать

на нижнем уровне двухуровневых итерационных методов декомпозиции областей. В силу существующего для большинства случаев изоморфизма сеточных шаблонов и портретов матричных строк анализ декомпозиции может проводиться одинаковым образом как в терминах сеточных графов, так и на языке матричных графов, в силу чего употребляются названия «алгебраическая декомпозиция» и «геометрическая декомпозиция». Более целесообразным и естественным выглядит реализация декомпозиции на этапе построения сетки, когда можно оперировать информацией о геометрических объектах расчетной области, топологическими связями и расстояниями, и д.р. Однако на практике зачастую при использовании библиотеки алгебраических решателей декомпозиция не задается, и в таких случаях фактически требуется построить по каким-то критериям блочную структуру СЛАУ, пользуясь только матричным CSR-форматом. Эта задача имеет достаточно высокую информационную и логическую сложность, особенно если ее требуется решить в параллельном, т.е. распределенном по различным процессорам, режиме.

Формирование пересекающихся сеточных подобластей и соответствующих расширенных СЛАУ осуществляется в два этапа. На первом определяются сбалансированные (с примерно равным числом узлов) подобласти без перехлеста, для чего могут использоваться инструментарии типа популярной библиотеки METIS [18] (в том числе существующей в параллельном варианте), осуществляющей операции над графами. При этом подобласти каким-либо образом упорядочиваются (это соответствует разбиению матрицы на блочные строки), и в соответствии с этим нумеруются узлы сетки (или матричные строки): сначала идут все узлы (строки) из первой подобласти (блочной строки), затем — из второй, и т.д. На втором этапе производится постепенное расширение подобластей по слоям соседних узлов, или фронтов: сначала к внутренним узлам подобласти присоединяются ближайшие внешние узлы, являющиеся непосредственными соседями к внутренним и которые образуют 1-й слой расширения, затем к ним аналогично присоединяются узлы 2-го слоя, и т.д. до заданного априори количества фронтов расширения.

В матричной технологии это осуществляется только на основе глобального CSR-формата, а результатом являются локальные матричные CSR-форматы для каждой из расширенной подобластей (при этом, естественно, требуется перенумерация матричных строк и столбцов соответствующих ненулевых элементов из глобальной упорядоченности в локальную). Если матрицы расширенных СЛАУ в подобластях формируются только один раз, то их правые части необходимо пересчитывать в околограничных узлах на каждой внешней итерации. Для этого требуется определять совокупности множеств узлов-доноров и узлов-акцепторов, которые передают информацию от своей подобласти к соседним и, наоборот, принимают данные.

Второй требующий решения технологический вопрос — как и в каких пространствах осуществлять внешний итерационный процесс? Наиболее прямой и естественный путь заключается в реализации крыловского метода на основе формулы (7), где u^n и f полные векторы с размерностью, равной порядку исходной СЛАУ (1), т.е. $O(h^{-3})$ в трехмерном случае, где h есть характерный шаг сетки. В этом случае метод FGMRES исполняется в кластерном варианте с помощью P MPI-процессов, что требует, в частности, дополнительных обменов при вычислении скалярных произведений векторов.

Однако существует и альтернативный подход, состоящий в проведении внешних итераций в пространстве следов, т.е. для векторов, определенных только на смежных границах пересекающихся подобластей [10]. В этом случае исходная СЛАУ формально редуциру-

ется путем исключения неизвестных, соответствующих внутренним узлам подобластей, и итоговая размерность системы понижается на порядок, т.е. до $O(h^{-2})$. При этом реализацию внешнего FGMRES можно осуществлять только на одном процессоре, используя распараллеливание ограниченными средствами OpenMP, но полностью избегая при этом коммуникационных потерь. Однако такая вычислительная технология имеет существенный недостаток — неизбежный дефицит оперативной памяти одного процессора при решении больших СЛАУ с масштабируемым параллелизмом, требующим формирования очень большого количества подобластей.

Что касается не вычислительных, а программных и информационных технологий ускорения параллельных процессов для рассматриваемых классов задач и алгоритмов, то здесь также имеются значительные резервы за счет оптимизации кода, организации развертки циклов, выбора режимов компиляции, профессионального использования командных возможностей систем MPI и OpenMP, и д.р. Конечный результат здесь в существенной степени зависит от искусства математика-программиста в синхронизации массовых расчетных и обменных операций, имея целью добиться максимум возможного в условиях ограниченных рамок маневрирования вычислительными потоками на фиксированных функциональных характеристиках аппаратных устройств МВС. Но так или иначе, человеческий фактор в существующих условиях еще играет далеко не последнюю роль.

3. Примеры технических требований к библиотекам алгебраических решателей

Программного обеспечения по вычислительной алгебре в мире существует достаточно много и давно, как самостоятельного, так и в составе пакетов прикладных программ, как коммерческого, так и общедоступного. Однако библиотек по итерационным параллельным алгоритмам решения больших разреженных СЛАУ имеется не так уж много. Помимо уже упоминавшейся библиотеки CUSP для GPU, можно назвать такие разработки, как Hypre [19], PETSc [20] и Saad Software [21]. Высокопроизводительные продукты для решения сверхбольших СЛАУ на суперкомпьютерах петафлопного масштаба еще ждут своего часа, так что мы рассмотрим некоторые технические требования к библиотеке алгебраических решателей, руководствуясь в первую очередь практическими требованиями, возникающими из актуальных проблем математического моделирования. Более того, мы будем исходить из целевой постановки о создании программного обеспечения, ориентированного на широкого круга пользователей и рассчитанного на длительный жизненный цикл с регулярным развитием функционального наполнения, а также адаптирующегося к эволюции компьютерных архитектур.

а. *Состав решаемых задач и алгоритмов.* Обладая определенным багажом знаний и технологий, или ноу-хау, заманчиво на единых принципах охватить СЛАУ различных типов: вещественных и комплексных, эрмитовых и неэрмитовых, положительно определенных и знаконеопределенных и д.р. Такую систематизацию можно продолжить до выделения специальных систем, для которых применимы сверхбыстрые алгоритмы.
Основа современных итерационных процессов — это предобусловленные методы в крыловских подпространствах. Однако в этих обоих направлениях имеется огромное разнообразие вариантов, и выше мы только коротко остановились на двухуровневых методах декомпозиции областей.

б. *Принципы организации интерфейса.* Традиционный вопрос в данном случае такой — делать ли алгебраический решатель в форме «черного ящика» или, наоборот, в качестве полностью открытого пользователю и настраиваемого на конкретные задачи инструмента? Возможны, конечно, и различные компромиссные «серые» варианты. Целесообразные решения, естественно, зависят от того, на какого типа пользователя рассчитан программный продукт. Наиболее реальной является ситуация, когда решатель нужен не в автономном варианте, а встраиваемый в уже существующий ППП.

Традиционная форма универсальной программной реализации итерационного метода крыловского типа такова: широкое использование BLAS-овских функций для векторных операций, а также открытость для внешних процедур умножения вектора на матрицу исходной СЛАУ и на предобуславливатель. Такое абстрагирование изначально закладывается в концепцию объектно-ориентированного программирования, например, в языке C++. Однако хорошо известно, что чрезмерный универсализм — это враг эффективности и экономичности.

Отсюда возникает немаловажный вопрос — как встраивать в широко-форматную библиотеку специальные сверхбыстрые алгоритмы для частного вида СЛАУ, которые не укладываются в общую вышеприведенную схему? Например, расчетная область или даже одна из подобластей могут допускать разделение переменных и, как следствие, применение быстрого преобразования Фурье (БПФ), которое очень жаль было бы запрещать к использованию.

в. *Проблема переиспользования программ.* Разработанное компьютерным сообществом прикладное программное обеспечение представляет огромный интеллектуальный потенциал, и умение его использовать дает заведомое преимущество разработчикам программного продукта. В рассматриваемых нами обстоятельствах это означает, в первую очередь, наличие первоклассных программ, реализующих прямые методы, которые можно и нужно использовать для решения СЛАУ в подобластях.

г. *Внутренняя структура и организация библиотеки.* Один из возможных способов построения библиотеки алгебраических решателей — это создание каталогизированного и систематизированного хранилища большого количества алгоритмов, из которого специальными конфигурационными средствами может формироваться конкретная версия продукта для определенных условий эксплуатации. Более примитивный подход — конкретная программа просто вынимается из хранилища (переписывается в файл) и отторгается. Третий, и наиболее продвинутый, способ существования прикладной библиотеки состоит в создании баз данных для типовых СЛАУ, для результатов их решения различными алгоритмами, а также в организации тренинга и обучения потенциальных пользователей.

Список технологических и примыкающих организационных вопросов по созданию библиотеки параллельных алгоритмов решения больших разреженных СЛАУ можно было бы значительно продолжить (многоязычность и многоверсионность, платформонезависимость, внутренние средства развития и адаптируемости, сопровождение, документируемость, лицензионность и д.р.). Частично эти вопросы были решены разработчиками библиотеки KRYLOV, которая находится на стадии опытной эксплуатации в ИВМиМГ СО РАН в авторском сопровождении.

Режим исполнения в библиотеке KRYLOV можно назвать полуавтоматическим, или в стиле «серого ящика». Внешний итерационный процесс по входному заданию пользователя

может выполняться или на корневом процессоре в пространстве следов, или в распределенном варианте. Для решения внутренних СЛАУ в подобластях допустимо использование как авторских реализаций различных крыловских процессов, так и прямой алгоритм PARDISO, а также ряд решателей из библиотеки CUSP NVIDIA. В распоряжении пользователя – различные методы декомпозиции с параметризованными размерами пересечений подобластей, типами итерируемых краевых условий на смежных границах, количество задействованных подобластей и соответствующих MPI-процессов, число вычислительных потоков на CPU, а также различные счетные параметры для возможного управления скоростью сходимости итераций. Предусмотрены также режимы выборы параметров по умолчанию без вмешательства пользователя.

Заключение

В работе рассмотрены актуальные математические и технологические проблемы решения сверхбольших плохо обусловленных СЛАУ, принципиально влияющие на эффективность параллельных программных реализаций алгоритмов для МВС сложной современной архитектуры. Основное положение заключается в комплексности тесно взаимосвязанных вопросов: обоснование, сходимость и оптимизация крыловских процессов для широкого многообразия типов и свойств матриц; алгебро-геометрические аспекты предобуславливания СЛАУ; алгоритмические и технологические особенности различных видов декомпозиции областей; основные подходы к ускорению крыловских итерационных методов на базе грубосеточной коррекции, дефляции и других приемов улучшения или расширения базисных векторов для подавления ошибки; программные реализации параллельных алгебраических решателей на вычислительных системах с многоуровневой распределенной и общей памятью.

Работа поддержана грантом РФФИ №11-01-00205, а также грантами Президиума РАН №15.9-4 и ОМН РАН №1.3.3-4.

Литература

1. Ильин, В.П. Методы и технологии конечных элементов / В.П. Ильин. — Новосибирск: Изд-во ИВМиМГ СО РАН, 2007.
2. Лебедев, В.И. Вариационные алгоритмы метода разделения области / В.И. Лебедев, В.И. Агошков.— М., Препр. ОВМ РАН; — 1983.—№ 54.
3. Bramble, J.H. Convergence estimates for product iterative methods with applications to domain decomposition / J. Bramble, J. Pasciak, J. Wang, J. Xu // Math. Comp. — 1991. — Vol. 57, № 195. — P. 1–21.
4. Ильин, В.П. Параллельные методы и технологии декомпозиции областей / В.П. Ильин // Вестник ЮУрГУ. Серия «Вычислительная математика и информатика». — 2012. — Т. 46, № 305. — С. 31–44.
5. Saad, Y. Iterative Methods for Sparse Linear Systems, Second Edition / Y. Saad. — SIAM, 2003.
6. Бутюгин, Д.С. Krylov: библиотека алгоритмов и программ для решения СЛАУ / Д.С. Бутюгин, В.П. Ильин, Е.А. Ицкович и др. // Современные проблемы математиче-

ского моделирования. Математическое моделирование, численные методы и комплексы программ. Сборник трудов Всероссийских научных молодежных школ. Ростов-на-Дону: Изд-во Южного федерального университета, 2009. — С. 110–128.

7. Ильин, В.П. Проблемы высокопроизводительных технологий решения больших разреженных СЛАУ / В.П. Ильин // Вычислительные методы и программирование. — М., МГУ. — 2009. — Т. 10, № 1. — С. 141–147.

8. Бутюгин, Д.С. Методы параллельного решения СЛАУ на системах с распределенной памятью в библиотеке Krylov / Д.С. Бутюгин, В.П. Ильин, Д.В. Перевозкин // Вестник ЮУрГУ. Серия «Вычислительная математика и информатика». — 2012. — Т. 47, № 306. — С. 5–19.

9. Intel Math Kernel Library from Intel. URL: `http://software.intel.com/en-us/intel-mkl` (дата обращения: 15.02.2013).

10. Ильин, В.П. Параллельные методы декомпозиции в пространствах следов / В.П. Ильин, Д.В. Кныш // Вычислительные методы и программирование. — 2011. — Т. 12, № 1. — С. 100–109.

11. Brezina, M. An improved convergence analysis of smoothed aggregation algebraic multigrid / M. Brezina, P. Vanek, P.S. Vassilevsky // Numer. Linear Algebra Appl. — 2012. — Vol. 19. — P. 441–469.

12. Farhat, C. FETI-DP: A dual-primal unified FETI method. Part I: A faster alternative to the two-level FETI method / C. Farhat, M. Lesoinne, P. LeTollei, K. Pierson, D. Rixen // Int. J. Numer. Math. Engrg. — 2001. — Vol. 50. — P. 1523–1544.

13. Кластер НКС-30Т: URL: `http://www2.sscc.ru/HKC-30T/HKC-30T.htm` (дата обращения: 15.02.2013).

14. Message Passing Interface at Open Directory Project: URL: `http://www.dmoz.org/Computers/Parallel_Computing/Programming/Libraries/MPI/` (дата обращения: 15.02.2013).

15. Малышкин, В.Э. Параллельное программирование мультикомпьютеров / В.Э. Малышкин, В.Д. Корнеев // Новосибирск: Изд. НГТУ, 2006.

16. CUDA Tools & Ecosystem: URL: `http://developer.nvidia.com/cuda-tools-ecosystem` (дата обращения: 15.02.2013).

17. Bell, N. CUSP: Generic parallel algorithms for sparse matrix and graph computations / N. Bell, M. Garland.// URL: `http://cusp-library.googlecode.com` (дата обращения: 15.02.2013).

18. Karypis, G. A fast and high quality multilevel scheme for partitioning irregular graphs / G. Karypis, V. Kumar // SIAM J. Sci. Comp. — 1999. — Vol. 20, № 1. — P. 359–392.

19. Hypre: URL: `http://acts.nersc.gov/hypre/` (дата обращения: 15.02.2013).

20. PETSc: Home Page: URL: `http://www.mcs.anl.gov/petsc/` (дата обращения: 15.02.2013).

21. Yousef Saad — SOFTWARE: URL: `http://www-users.cs.umn.edu/~saad/software/` (дата обращения: 15.02.2013).

Ильин Валерий Павлович, д.ф.-м.н., г.н.с. Института вычислительной математики и математической геофизики СО РАН, профессор Новосибиского Государственного Университета (Новосибирск, Российская Федерация), ilin@sscc.ru

---

ON THE QUESTIONS OF PARALLELIZED KRYLOV'S ITERATIVE METHODS

V.P. Il'in, Institute of Computational Mathematics and Mathematical Geophysics of Siberian Branch of the Russian Academy of Sciences (Novosibirsk, Russian Federation)

Mathematical questions of various computational technologies of parallelelized iterative processes of Krylov's type for solving large sparse symmetric and non-symmetric SLAEs, obtained in grid approximations of multi-dimensional boundary value problems for PDEs, are considered. Example are presented by finite approximations in gas-hydrodynamical applications, where five unknowns in each node are defined and corresponding SLAEs have small-block structure. The base of used algorithms is flexible generalized minimal residual, FGMRES, method with dynamical preconditioners of additive type, which presents an upper level of two-step iterarive Swartz algorithm.

High performance of algebraic solvers is provided by using different approaches: domain decompositions of various topologies, boundary conditions and sizes of subdomain overlapping, coarse grid correction, deflation and aggregation, and incomplete factorizations of matrices. The unified formulations of using algorithms as well as the questions of computational efficiency and scalable parallelization at the geterogenous supercomputers are described. The examples of technical requirements for peculiarities of program implementation of the libraries of parallel algorithms for solving systems of linear algebraic equation, are presented.

Keywords: iterative methods, Krylov subspaces, preconditioned matrices, domain decomposition, parallel algorithms, program and computational technologies.

References

1. Il'in V.P. Metody i tekhnologii konechnykh elementov [Finite element methods and technologies]. Novosibirsk, NSC Publ., 2007.
2. Lebedev V.I., Agoshkov V.I. Variazionnyi method dekompozitsii oblastei [Variational domain decomposition method]. Moscow, DCMRAS, preprint No 54, 1984.
3. Bramble J.H., Pasciak J.E., Wang J., Xu J. Convergence estimates for product iterative methods with applications to domain decomposition. Math. Comp. 1991. Vol. 57, No 195. P. 1–21.
4. Il'in V.P. Parallelnye methody i tehnologii dekompozitsii oblastei [Parallel domain decomposition methods and technologies]. Vestnik YUURGU. Seriya "Vychislitelnaya matematika i informatika"[Bulletin of South Ural State University. Seriers: Computational Mathematics and Software Engineering]. 2012. Vol. 46, No 305. P. 31–44.
5. Saad Y. Iterative Methods for Sparse Linear Systems, Second Edition. SIAM, 2003.
6. Butyugin D.S., Il'in V.P., Itskovich E.A. et all. Krylov: biblioteka algoritmov i program dlya teshenia SLAU.[Krylov: the library of algorithms and programs for solving SLAEs]. Modern problems of math. modeling. Rostov-Don, YUFU Publ., 2009. P. 110–128.
7. Il'in V.P. Problemy vysokoproizvoditelnyh tehnologiy reshenia bolshih redkih SLAU [Problems of high performance technologies of solving large sparse SLAEs] Vychislitelnye

metody i programmirovanie [Computational methods and programming]. 2009. Vol. 10, No 1. P. 141–147.

8. Butyugin D.S., Il'in V.P., Perevozkin D.V. Metodu parallelnogo resheniya SLAU na systemah s raspredelennoi pamyatyu [Methods of parallel solving SLAEs on the systems with distributed memory]. Vestnik YUURGU. Seriya "Vychislitelnaya matematika i informatika"[Bulletin of South Ural State University. Seriers: Computational Mathematics and Software Engineering]. 2012. Vol. 47, No 306. P. 5–19.
9. Intel (R) Math Kernel Library from Intel. URL: http://software.intel.com/en-us/intel-mkl (accessed 12 February 2013).
10. Il'in V.P., Knysh L.V. Parallelnye metody dekompozitsii v prostranstve sledov [Parallel domain decomposition methods in trance subspaces]. Computational methods and programming. 2011. Vol. 12, No 1. P. 100–109.
11. Brezina M., Vanek P., Vassilevsky P.S. An improved convergence analysis of smoothed aggregation algebraic multigrid. Numer. Linear Algebra Appl. 2012. Vol. 19. P. 441–469.
12. Farhat C., Lesoinne M., LeTollei P., Pierson K., Rixen D. FETI-DP: A dual-primal unified FETI method. Part I: A faster alternative to the two-level FETI method. Int. J. Numer. Math. Engrg. 2001. Vol. 50. P. 1523–1544.
13. Cluster HKC-30T: URL: http://www2.sscc.ru/HKC-30T/HKC-30T.htm (accessed 12 February 2013).
14. Message Passing Interface at Open Directory Project: URL: http://www.dmoz.org/Computers/Parallel_Computing/Programming/Libraries/MPI/ (accessed 12 February 2013).
15. Malyshkin V.E., Korneev V.D. Parallelnoe programmirovanie multikompyuterov [Parallel programming the multi-computers]. Novosibirsk, NSTU Publ., 2006, 310 p.
16. CUDA Tools & Ecosystem: URL: https://developer.nvidia.com/cuda-tools-ecosystem (accessed 12 February 2013).
17. Bell N., Garland M. CUSP: Generic parallel algorithms for sparse matrix and graph computations: URL: http://cusp-library.googlecode.com (accessed 12 February 2013).
18. Karypis G., Kumar V. A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM J. Sci. Comp. 1999. Vol. 20, No 1. P. 359–392.
19. Hypre: URL: http://acts.nersc.gov/hypre/ (accessed 12 February 2013).
20. PETSc: Home Page: URL: http://www.mcs.anl.gov/petsc/ (accessed 12 February 2013).
21. Yousef Saad – SOFTWARE: URL: http://www-users.cs.umn.edu/~saad/software/ (accessed 12 February 2013).

Поступила в редакцию 14 марта 2013 г.