РЕЗУЛЬТАТЫ ВЫЧИСЛИТЕЛЬНЫХ ЭКСПЕРИМЕНТОВ РЕШЕНИЯ ДВУМЕРНОЙ УПРУГО-ПЛАСТИЧЕСКОЙ ЗАДАЧИ ИТЕРАЦИОННЫМИ МЕТОДАМИ НА КЛАСТЕРНОЙ СИСТЕМЕ

И.П. Демешко[1], Е.Н. Акимова[2], А.В. Коновалов[1]

[1]Институт машиноведения Уральского отделения Российской академии наук, Екатеринбург, 620219, Комсомольская 34

[2] Институт математики и механики Российской академии наук, Екатеринбург, 620219, С.Ковалевской, 16

Аннотация

В работе представлены результаты применения параллельных алгоритмов решения методом конечных элементов двумерных упруго-пластических задач с большими пластическими деформациями на примере решения задачи сжатия цилиндра из упруго-пластического изотропного и изотропно-упрочняемого материала плоскими плитами на многопроцессорной вычислительной системе МВС-1000/32.

COMPUTING EXPERIMENTS REZULTS OF SOLUTION OF 2D ELASTO-PLASTIC PROBLEMS BY THE ITERATION METHODS ON THE CLUSTER SYSTEM

Results of applying of parallel algorithms for the solution of 2D elasto-plastic problems with large plastic strains using finite element method at the multiprocessor computing system MVS-1000/32 are presented. As an example, we consider the problem of cylinder compression from elasto-plastic isotropic and isotropic-strengthened material by flat plates

1. ВВЕДЕНИЕ

Параллельные вычисления на МВС позволяют ускорить решение сложных задач, требующих больших объемов вычислений за счет распределения расчетов между процессорами с помощью библиотек параллельных вычислений. В работе описаны результаты применения параллельных вычислений на примере решения двумерной упруго-пластической задачи сжатия цилиндра плоскими плитами на МВС-1000/32.

2. ОПИСАНИЕ АЛГОРИТМОВ И РЕЗУЛЬТАТОВ

Решение рассмотренной задачи основывается на принципе виртуальной мощности в скоростной форме [1], уравнение которого имеет вид

$$\int_V \left(\sigma + \Delta t\dot{\sigma}\right)\cdot\cdot\nabla h\, dV + \int_\Sigma \left(P + \Delta t\dot{P}\right)\cdot h d\Sigma = 0 \qquad (1)$$

со следующими определяющими соотношениями, полученными в работе [2]:

$$\dot{\sigma} = \lambda\dot{\Theta}I + 2(\lambda\Theta + \mu)D - \nabla \mathrm{v}\cdot\sigma - \sigma\cdot\nabla \mathrm{v}^T - JbS,$$
$$S = \sigma - \sigma_0 I, \quad \sigma_0 = K\Theta, \quad K = \lambda + 2/3\,\mu,$$
$$D = 0.5(\nabla \mathrm{v} + \nabla \mathrm{v}^T),$$
$$F(S,k) = 0.5S\cdot\cdot S - k^2 = 0, \qquad (2)$$
$$b = \frac{\left[\left[\mu\left(1-\frac{2}{3}\Theta\right)S - S\cdot S\right]\cdot\cdot D - \frac{\mu}{3}\dot{\Theta}\, S\cdot\cdot I\right]}{\left[k^2\left(1+\frac{1}{\mu}\frac{dk}{d\chi}\right)\right]}.$$

Здесь σ – тензор напряжений Коши; P – плотность поверхностных сил; Δt – промежуток времени для шага приращения нагрузки; h – вариация кинематически допустимых полей скоростей; ∇ – набла-оператор; V, Σ – объем и поверхность тела соответственно; dV, $d\Sigma$ – элементы объема и площади поверхности тела соответственно; λ, μ - коэффициенты Ламе; I – единичный тензор; точкой и двумя точками обозначено соответственно скалярное и двойное скалярное произведение тензоров; точкой сверху обозначена полная производная по времени; ∇V – градиент скорости перемещений; D – тензор скоростей деформаций; σ_0 – среднее нормальное напряжение; K – объемный модуль упругости; Θ – относительное изменение индивидуального объема бесконечно малой частицы среды; k – напряжение текучести; $J = 0$ при $F < 0$ или при $F = 0$, $S\cdot\cdot\dot{S} \le 0$; $J = 1$ при $F = 0$, $S\cdot\cdot\dot{S} > 0$; χ – параметр упрочнения, в силу малости упругих деформаций $\dot{\chi} = \sqrt{2D\cdot\cdot D}$.

На контакте с плитами приняли закон трения Кулона. Нагрузку в виде перемещения плиты прикладывали малыми шагами Δh. Шаг Δh выбирали так, чтобы отношение $\Delta h/h$ (h - высота цилиндра) не превышало предела упругости по деформации, в нашем случае 0.002, что обеспечивало устойчивость вычислительной процедуры. Величину относительного сжатия цилиндра приняли равной 0.5. На каждом шаге нагрузки равенство (1) с помощью конечно-элементной аппроксимации сводится к СЛАУ:

$$Ax = b, \qquad (3)$$

где A, x, b - соответственно матрица, вектор решения и вектор правой части системы. Матрица A имеет ленточный вид.

Решение задачи сжатия цилиндра на шаге нагрузки состоит из трех основных этапов:

1. подготовка матрицы A,
2. решение СЛАУ,
3. вычисление напряженно-деформированного состояния в конце шага нагрузки.

Этап 1 выполняется один раз, а этапы 2 и 3 – десять раз, поскольку выполнение условия текучести Мизеса (2) достигается итерационно по шагам нагрузки. Анализ показал, что на шаге нагрузки время вычисления этих этапов с учетом итераций составляет соответственно 15%, 45% и 40%. С увеличением размерности

матрицы процентное отношение времени решения СЛАУ ко времени решения всей задачи увеличивается и, следовательно, уменьшается процентное отношение этапов подготовки матрицы и вычисления напряженно-деформированного состояния в конце шага нагрузки.

Распараллеливание и численную реализацию задачи выполнили на многопроцессорной системе МВС-1000/32, установленной в ИММ УрО РАН. Использовали язык Си и библиотеку параллельного программирования MPI[3]. МВС-1000/32 – многопроцессорный вычислительный комплекс кластерного типа, имеющий 32 процессора.

Организацию доступа к МВС осуществили через протокол SSH. SSH является сетевым протоколом прикладного уровня для удалённого управления операционной системой. Он безопасно передает в незащищенной среде большинство других сетевых протоколов, что позволяет удаленно работать на компьютере через командную оболочку и посылать данные на удаленный компьютер. Для работы с командной строкой МВС использовали программу PuTTY, которая является свободно распространяемым клиентом для протоколов SSH, Telnet, rlogin и TCP. Программу WinSCP применяли как самостоятельное приложение для копирования и редактирования файлов, находящихся на МВС. WinSCP является свободным графическим клиентом протоколов SFTP и SCP, предназначенным для Windows. Он обеспечивает защищенное копирование файлов между компьютером и серверами, поддерживающими эти протоколы. Взаимодействие между двумя частями программы осуществляли путем обмена файлами по сети. Для сохранения безопасности доступа к МВС передачу файлов производили через компоненты набора сетевых инструментов OpenSSH.

Параллельные алгоритмы разработали для каждого из перечисленных этапов решения задачи. Алгоритм распараллеливания этапа формирования матрицы A основан на распределении конечных элементов между процессорами. Общее количество конечных элементов n делится на число процессоров m и полученное значение округляется в сторону большего целого. Если остаток от деления n на m не равен нулю, то на процессор с наибольшим порядковым номером будет приходиться $n/m - m$ элементов, что на $m-1$ элемент меньше, чем на остальные процессоры. Каждый процессор независимо от других вычисляет локальные матрицы жесткости для своей группы конечных элементов. После окончания вычислений все процессоры обмениваются между собой заполненными локальными матрицами жесткости. После обмена данными процессоры независимо формируют свои копии глобальной матрицы жесткости по информации о соотношении номеров переменных в локальных и глобальной матрицах жесткости. Алгоритм реализован с помощью библиотеки MPI.

Для проведения вычислительного эксперимента применения этого алгоритма на МВС выполняли решение всей задачи с распараллеливанием этапа формирования матрицы жесткости. При этом на каждом процессоре размещалась своя копия всей вычислительной программы и всего массива информации. Решение рассматриваемой задачи требует формирования и обработки больших объемов данных и выполнения трудоемких вычислений. Из-за ограничения оперативной памяти на процессорах не удалось провести эксперименты на задаче с размерностью сетки больше 5x5.

На рис.1 представлены результаты распараллеливания на шаге нагрузки отдельных этапов решения задачи сжатия цилиндра для случая разбиения продольного сечения цилиндра сеткой 5x5. Здесь m - число процессоров, за ускорение a принято отношение времени решения задачи на одном процессоре ко времени решения на нескольких процессорах.

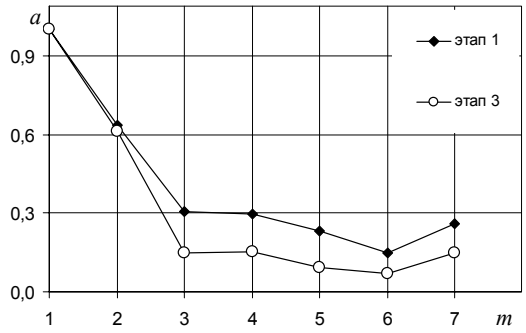

Рис.1. Зависимость ускорения a параллельных алгоритмов для этапов 1 и 3 решения задачи на шаге нагрузки от числа процессоров m при использовании технологии MPI

Результаты распараллеливания этапа 1 показали увеличение времени решения задачи (уменьшение значения ускорения a) с ростом числа процессоров m. Это объясняется тем, что из-за небольшой размерности сетки трудоемкость обмена данными между процессорами больше, чем вычислительная трудоемкость частей задачи, расположенных на отдельных процессорах.

Алгоритм вычисления напряженно-деформированного состояния в конце шага нагрузки так же, как алгоритм формирования матрицы жесткости, основан на распределении вычислений между процессорами по числу обрабатываемых конечных элементов. Каждый процессор независимо от других вычисляет параметры напряженно-деформированного состояния для своей части конечных элементов. Алгоритм реализован с помощью библиотеки MPI. Для проведения вычислительного эксперимента применения этого алгоритма на МВС выполняли решение всей задачи с распараллеливанием только третьего этапа.

Аналогично параллельному алгоритму этапа формирования матрицы этап вычисления напряженно-деформированного состояния в конце шага нагрузки показал увеличение времени вычисления с ростом числа процессоров, что видно из рис. 1. Это так же объясняется тем, что эксперимент был проведен на задаче с небольшой размерностью сетки (5x5).

Для решения СЛАУ (3) использовали итерационные методы [4] со следующими рекуррентными соотношениями:

1. МПИ:

$$z^{k+1} = z^k - \frac{1}{\lambda_{\max}}\left[\left(A+\alpha_k E\right)z^k - b\right],\ z_0 = 0,$$

где $\lambda_{\max}$ - максимальное собственное значение, E - единичная матрица, α_k - параметр регуляризации.

2. ММН:

$$z^{k+1} = z^k - \frac{\left(A\left(Az^k - b\right),\ Az^k - b\right)}{\left\| A\left(Az^k - b\right)\right\|^2}\left(Az^k - b\right),\quad z_0 = 0.$$

3. МНС:

$$z^{k+1} = z^k - \frac{\left\| A^T Az^k - A^T b\right\|^2}{\left\| A\left(A^T Az^k - A^T b\right)\right\|^2} A^T\left(Az^k - b\right), z_0 = 0.$$

4. МСГ [5]:

$$z^1 = z^0 - \frac{\left\| r^0\right\|^2 \cdot \left(B^k z^0 - b\right)}{\left(B^k r^0, r^0\right)}\ ,\ r^0 = B^k z^0 - b,$$

где $z^{k+1} = z^k - \gamma_k\left(B^k z^k - b\right) + \beta_k\left(z^k - z^{k-1}\right)$,

$$\gamma_k = \frac{\left\| r^k\right\|^2\left(B^k p^k, p^k\right) - \left(r^k, p^k\right)\left(B^k r^k, r^k\right)}{\left(B^k r^k, r^k\right)\left(B^k p^k, p^k\right) - \left(B^k r^k, p^k\right)},$$

$$r^k = B^k z^k - b,$$

$$\beta_k = \frac{\left\| r^k\right\|^2\left(B^k r^k, p^k\right) - \left(r^k, p^k\right)\left(B^k r^k, r^k\right)}{\left(B^k r^k, r^k\right)\left(B^k p^k, p^k\right) - \left(B^k r^k, p^k\right)},$$

$$p^k = z^k - z^{k-1},\ \ k = 0, 1, 2\ldots$$

В приведенных формулах k – номер итерации.

Алгоритм распараллеливания решения СЛАУ(3) итерационными методами основан на преобразовании ленточной матрицы в вертикальную полосу и разбиении ее горизонтальными полосами на m блоков. При этом вектор решения и вектор правой части СЛАУ разбивается на m частей так, чтобы $n = m \cdot L$, где n – размерность системы уравнений, m – число процессоров, L – число строк в одном блоке. Каждый процессор вычисляет свою часть вектора решения и передает вычисленные значения всем остальным процессорам.

Условием остановки итерационного процесса является: $\|Ax - b\|/\|B\| \le \varepsilon$, где ε – заданная точность решения. Выбор ε определялся из соотношения: $(\|x_Г\| - \|x_И\|)/\|x_Г\|$, где $\|x_Г\|$ – норма решения, полученного методом Гаусса, $\|x_И\|$ – норма решения, полученного итерационным методом. Из вычислительных экспериментов выбрали $\varepsilon \le 0.1$.

В качестве начального приближения решения СЛАУ(3) принимали вектор ее правой части. На каждой следующей итерации за начальное приближение брали решение системы, полученное на предыдущей итерации.

Для проведения вычислительного эксперимента на МВС запускали программу решения СЛАУ разработанными параллельными алгоритмами. При запуске программа считывала всю заранее подготовленную информацию о СЛАУ. Поэтому, оперативной памяти процессоров МВС было достаточно для проведения эксперимента на матрицах, соответствующих размерности сетки до 60x60.

Результаты решения СЛАУ(3) этими методами для матрицы с шириной ленты 171 и количеством переменных 3362, полученной для разбиения области сеткой 40x40, приведены на рис.2. Видно, что методы минимальных невязок, простой итерации, наискорейшего спуска показали близкие результаты при одинаковых условиях решения задачи. Метод сопряженных градиентов требует большее число итераций по сравнению с остальными рассмотренными методами и большее время для решения СЛАУ.

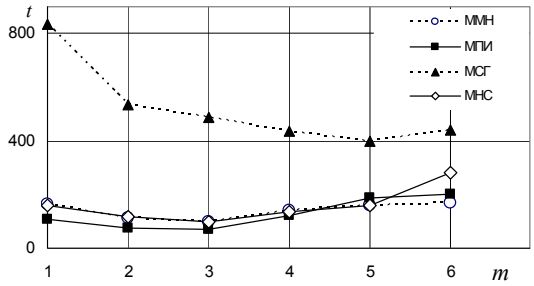

Рис.2. График зависимости времени вычисления t СЛАУ параллельными итерационными алгоритмами от числа процессоров m

На рис.3 представлена зависимость ускорения a параллельных алгоритмов решения СЛАУ(3) итерационными методами от числа процессоров m для сеток разной размерности. Результаты приведены для случая решения СЛАУ методом простой итерации, так как графики ускорения параллельных алгоритмов ММН, МПИ и МНС расположены близко друг к другу и показывают наилучшие результаты при решении рассматриваемой задачи.

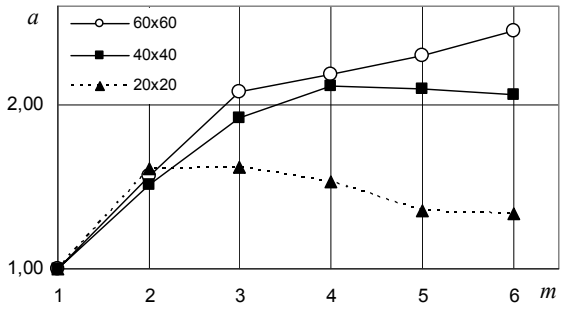

Рис.3. Зависимость ускорения a параллельного алгоритма решения СЛАУ методом простой итерации от числа процессоров m для сеток разной размерности.

Для всех рассмотренных итерационных методов с увеличением размерности матрицы A эффективность распараллеливания возрастает. Это объясняется тем, что при увеличении размерности сетки вычислительная трудоемкость для части матрицы, находящейся на одном процессоре, увеличивается. При этом объем передаваемых данных между процессорами увеличивается незначительно. В результате отношение времени вычислений к времени передачи данных растет с увеличением размерности матрицы. Следовательно, с увеличением размерности матрицы можно добиться большего ускорения на большем числе процессоров при распараллеливании итерационных алгоритмов.

ЗАКЛЮЧЕНИЕ

При решении двумерных упруго-пластических задачах с большими пластическими деформациями использование разработанных параллельных алгоритмов

решения СЛАУ(3) итерационными методами позволяет сократить время решения задачи при использовании МВС кластерного типа. Для эффективной реализации параллельных алгоритмов этапов формирования матрицы жесткости и вычисления напряженно-деформированного состояния на шаге нагрузки на МВС кластерного типа необходимо разделить информацию, связанную с конечными элементами, по процессорам в соответствии с разделением конечно-элементных вычислений между ними.

СПИСОК ОБОЗНАЧЕНИЙ

МВС – многопроцессорная вычислительная система;
СЛАУ – система линейных алгебраических уравнений;
ИММ – Институт математики и механики;
ММН - метод минимальных невязок;
МПИ - итеративно регуляризованный метод простой итерации;
МНС - метод наискорейшего спуска;
МСГ - метод сопряженных градиентов;

СПИСОК ЛИТЕРАТУРЫ

1. Поздеев А. А., Трусов П.В., Няшин Ю.И. Большие упруго-пластические деформации. М: Наука, 1986, 232с.
2. Коновалов А. В. Определяющие соотношения для упругопластической среды при больших пластических деформациях // Известия РАН. Механика твердого тела. 1997. № 5. С. 139-149.
3. MPI: A Message-Passing Interface Standard. Message Passing Interface Forum. – Version 1.1. 1995. – http://www-unix.mcs.anl.gov/mpi
4. Васин В.В., Ерёмин И.И. Операторы и итерационные процессы Фейеровского типа. Теория и приложения. – Екатеринбург, 2005. 210с.
5. Фадеев В.К., Фадеева В.Н. Вычислительные методы линейной алгебры. – Москва: Гос. издат. физико-математической литературы, 1963, 734с.