Parallel algorithm of the solution of boundary problem for system of the 1-st order ordinary differential equations

Authors: T.Davitashvili (TSU (GE)),
H.Meladze (GTU(GE)),
V.Saakyan (IIAP NAS RA (AM)),
P.Tsereteli (SANGU (GE))

- Введение
- Постановка задачи
- Описание итерационного метода
- Алгоритм решения задачи
- Реализация алгоритма для параллельной системы
- Результаты численных экспериментов

Введение

Параллельные вычислительные системы:

- Системы с общей памятью
  - SMP
  - NUMA
- Системы с распределенной памятью
- Графические процессоры для общих вычислений (GPGPU)

Введение

Технологии параллельного программирования:

- OpenMP
- MPI
- PVM
- MPP Fortran
- HP Fortran
- CUDA-C, CUDA-Fortran
- Linda

Постановка задачи

Рассмотрим задачу нахождения такого значения $\boldsymbol{\lambda}^*$ параметра $\boldsymbol{\lambda}$ и такой вектор-функций $X^*(t,\lambda) = (x^1(t,\lambda), \ldots, x^N(t,\lambda))$ которые удовлетворяют системе уравнений

$$\frac{dX}{dt} = F(t, X(t,\lambda), \lambda), \quad t \in (0,1) \qquad (1)$$

и следующим условиям

$$X(0,\lambda) = \Psi(\lambda), \quad x^1(1,\lambda) = 0 \qquad (2)$$

Постановка задачи

$$\lambda \in (\Lambda_0, \Lambda_1)$$

$$F(t, X, \lambda) = \left\{ f^1(t, X(t, \lambda), \lambda), \ldots, f^N(t, X(t, \lambda), \lambda) \right\}$$

$$\Psi(\lambda) = \left\{ \psi^1(\lambda), \ldots, \psi^N(\lambda) \right\}$$

$$f^i : [0,1] \times R^N \times (\Lambda_0, \Lambda_1) \to R^1$$

$$\psi^i : (\Lambda_0, \Lambda_1) \to R^1$$

$$i = \overline{1, N}$$

Постановка задачи

Эквивалентная задача

$$\frac{dX}{dt} = F(t, X(t,\lambda),\lambda), \quad X(0,\lambda) = \Psi(\lambda)$$

с уравнением относительно λ:

$$x^1(1,\lambda) = \psi^1(\lambda) + \int_0^1 f^1(\tau, X(\tau,\lambda),\lambda)d\tau = 0$$

Описание итерационного метода

Пусть, *f(x)* – функция одной вещественной переменной
Существует интервал *I=(a,b)*, в котором уравнение *f(x)=0*
Имеет единственный корень $\bar{x}$ и $f(x) \in C^k(I)$, а также $f'(x) \neq 0$, когда $x \in I$.

Поскольку $f(\bar{x}) = 0$, найдется такая окресность I^* точки 0, где существует функция $g = f^{-1}$ и $g \in C^k(I^*)$

Рассмотрим последовательность N-мерных векторов

$$\mathrm{X}^{(\mathrm{i})} = (x_1^{(i)}, x_2^{(i)}, \cdots, x_N^{(i)}), \quad i = 0,1,2,\cdots$$

Кажда компонента которых есть приближенное значение корня $\bar{x}$.

Описание итерационного метода

Пусть, задано начальное приближение

$$\vec{X}^{(0)}=(x_1^{(0)},\cdots,x_{N-1}^{(0)},x_N^{(0)})$$

а также значения функций и производных:

$$f(x_1^{(0)})=y_1^{(0)},\ \cdots\quad f(x_N^{(0)})=y_N^{(0)},$$

$$f'(x_1^{(0)}),\ \cdots\quad f'(x_N^{(0)}),$$

$$\cdots\cdots\cdots\cdots\cdots\cdots$$

$$f^{(\alpha_1-1)}(x_1^{(0)}),\ \cdots\quad f^{(\alpha_N-1)}(x_N^{(0)}),$$

$$(\alpha_j\le k+1,\quad j=\overline{1,N}).$$

Описание итерационного метода

Можно найти и производные обратной функции *x=g(y)* в этих точках

$$x' = g' = 1/y', \quad x'' = g'' = -y''/(y')^3.$$

$$\frac{d^k x}{dy^k} = g^{(k)}(y) = \frac{\chi_k}{(y')^{2k-1}}$$

$$\chi_1 = 1, \quad \chi_2 = -y''$$

$$\chi_{n+1} = \frac{d}{dx}(\chi_n)y' - (2n-1)\chi_n y''.$$

Описание итерационного метода

Очевидно, что $\bar{x} \equiv g(0)$

Для нахождения $\bar{x}$ заменим *g(y)* интерполяционным полиномом Эрмита на каком-нибудь подмножестве множества $\left\{y_j^{(i)}\right\}_{j=1}^{N}$ и вычислим значения в точке 0.

$$H_j^{(i)}(y_s^{(i)}) = x_s^{(i)},$$

$$\left[H_j^{(i)}(y_s^{(i)})\right]' = \left[x_s^{(i)}\right]',$$

$$\ldots\ldots\ldots\ldots$$

$$\left[H_j^{(i)}(y_s^{(i)})\right]^{(\alpha_{js}-1)} = \left[x_s^{(i)}\right]^{(\alpha_{js}-1)}, \quad s \in A_j, \quad j = \overline{1,N}$$

где A_j есть некотрое подмножество индексов $A=\{1,2,\ldots,N\}$

Описание итерационного метода

Построим итерационый алгоритм вычисления j-компоненты вектора $\vec{X}^{(i+1)}$:

$$x_j^{(i+1)} = H_j^{(i)}(0), \quad j = \overline{1, N}$$

Алгоритм параллельный.

Описание итерационного метода

Используем метод для решения задачи

$$y'' = y^2 - 1, \quad x \in (0,1), \quad y(0) = 0, \quad y(1) = 1.$$

Перепишем в виде:

$$x = \int_0^y (2(c + \frac{y^3}{3} - y))^{-1/2} dy \qquad y'(0) = \sqrt{2c}.$$

Учитывая граничные условия получаем

$$\phi(c) = 1 - \int_0^1 (2(c + \frac{y^3}{3} - y)^{-1/2} dy = 0.$$

Описание итерационного метода

Итерационная формула имеет вид

$$x_j^{(i+1)} = \frac{1}{(y_{j+1} - y_j)^2} \cdot \sum_{k=0}^{1} y_{j+k}^2 \left(x_{j+1-k} + \frac{2x_{j+1-k} y_{j+1-k}}{(-1)^k (y_{j+1} - y_j)} - \frac{y_{j+1-k}}{f'(x_{j+1-k})} \right)$$

N=6 – число начальных приближений

M=2- число узлов в полиноме Эрмита

$$A_j = \{j, j+1\}, \quad j = \overline{1,5}, \quad A_6 = \{6,1\}$$

Алгоритм решения задачи

$$\frac{dX}{dt} = F(t, X(t,\lambda),\lambda), \quad X(0,\lambda) = \Psi(\lambda)$$

Виберем начальные приближения: $\lambda_1^{(0)}, \lambda_2^{(0)}, \lambda_3^{(0)}$

Получаем последовательность задач:

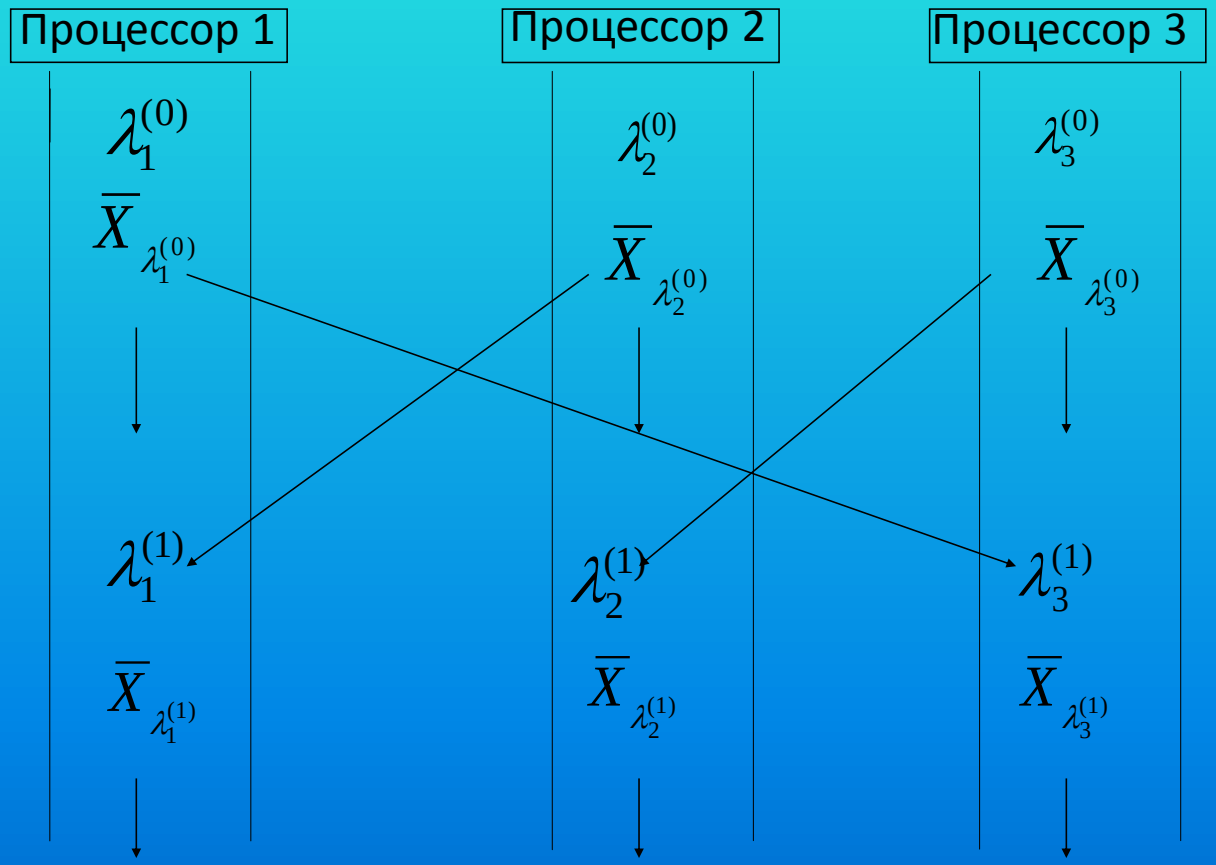
$$\frac{dX(t,\lambda_j^{(i)})}{dt} = F(t, X(t,\lambda_j^{(i)}),\lambda_j^{(i)}), \quad X(0,\lambda_j^{(i)}) = \Psi(\lambda_j^{(i)})$$

$$\lambda_j^{(i+1)} = H_j^{(i)}(0)$$

$$\lambda_j^{(i+1)} = \frac{1}{(z_{j+1} - z_j)^2} \cdot \sum_{k=0}^{1} z_{j+k}^2 \left(\lambda_{j+1-k} + \frac{2\lambda_{j+1-k} z_{j+1-k}}{(-1)^k (z_{j+1} - z_j)} - z_{j+1-k} \cdot \varphi'(z_{j+1-k}) \right)$$

$\lambda_1^{(0)}$

Алгоритм решения задачи



Реализация алгоритма для параллельной системы

- Ввод конечных точек отрезка (*a* и *b*), на котором задача имеет решение и итерационный процесс сходится.
- Кроме точек *a* и *b,* определяются точки $\lambda_1,\ldots,\lambda_{p-2}$ в интервале *(a,b)*, где p – число используемых процессоров. Получается *p* пар точек $(a,\lambda_1),(\lambda_1,\lambda_2),\ldots,(\lambda_{p-2},b)$ Каждая пара будет находится на отдельном процессоре. Обозначим их через через λ_1 и λ_2 .
- Решение задач Коши для λ_1 и λ_2 на каждом процессоре одновременно.

Реализация алгоритма для параллельной системы

- Решение задачи Коши (16)-(17) для вычисления производных функции φ
- Вычисление λ, согласно итерационной формуле, приведенной в разделе 3, на всех процессорах одновременно
- Если хотя бы на одном процессоре заданная точность достигнута, процесс заканчивется
- Если заданная точность не достигнута ни на одном процессоре от j-го процессора вычисленная λ пересылается $(j-1)$-му процессору по кольцеобразной схеме. Принятое значение λ будет прсивоено к λ_2

Результаты численных экспериментов

- Процесс сходится за меньшее число итерации, чем при методе Ньютона
- Интервал сходимости больше
- Увеличение узлов и тем самым увеличение количества процессоров заметно улучшаеткачество метода (увеличивается интерграл искорость сходимости

Спасибо за внимание