

Алгоритм поиска ближайших соседей в методе сглаженных частиц и его параллельная реализация

К. Е. АФАНАСЬЕВ, Р. С. МАКАРЧУК, А. Ю. ПОПОВ
Кемеровский государственный университет, Россия
e-mail: keafa@kemsu.ru, mak@kemsu.ru, a_popov@kemsu.ru

This work addresses an optimization of the SPH program realization. The nearest neighbor search algorithm and its parallel implementation are discussed. The results on acceleration of the algorithms taken from numerical simulations of real problems are presented.

1. Метод сглаженных частиц

Основная идея метода SPH состоит в дискретизации сплошной среды набором частиц, которые движутся в потоке жидкости, заполняющей область Ω , причем используется интегральное представление характеристик течения в виде

$$A(r) = \int_{-\infty}^{\infty} A(r')\delta(r - r')dr',$$

где δ — дельта-функция Дирака.

Далее вышеприведенный интеграл аппроксимируется интегралом следующего вида:

$$A(r) = \int_{\Omega} A(r')W(r - r', h)dr',$$

с весовой функцией W , которую часто называют функцией ядра, а сами интегралы заменяются конечной суммой [1]:

$$A(r) \approx \sum_{i=1}^n A_i \frac{m_i}{\rho_i} W(r - r_i, h),$$

где r_i , m_i и ρ_i — радиус-вектор, масса и плотность i -й частицы соответственно; n — количество соседних к i -й частиц. Две частицы i и j называются соседними или взаимодействующими друг с другом, если расстояние между ними не превосходит $(h_i + h_j)$. Величина $(h_i + h_j)$ является носителем функции ядра W , а h_i называется сглаживающей длиной i -й частицы и определяет радиус ее взаимодействия с окружением. В качестве веса W обычно используют полиномиальные сплайны.

2. Алгоритм поиска ближайших соседей

Особенностью метода SPH является то, что на каждом временном шаге для каждой частицы необходимо определять частицы, с ней взаимодействующие.

Временная сложность алгоритма, основанного на полном переборе частиц, требует порядка $O(N^2)$ операций, где N — число частиц области. Поэтому необходима реализация более эффективного алгоритма поиска взаимодействующих частиц.

Пусть требуется найти ближайших соседей для k -й частицы (рис. 1, *а*). Алгоритм поиска, рассматриваемый в данной работе, заключается в следующем. Строится сетка, покрывающая область течения. Размер ячейки сетки выбирается следующим образом: ячейка сетки является квадратом со стороной $2h_{\max}(1 + \varepsilon)$, где h_{\max} — максимальная сглаживающая длина из всех частиц, ε — некоторый параметр (при расчетах использовалось $\varepsilon = 0.01$). Подобный способ позволяет перебирать частицы только данной и соседних ячеек (рис. 1, *б*).

Тестовые расчеты проводились на однопроцессорной системе Athlon 2000+ / 512 Мб RAM / Windows XP SP2 в программной среде Fortran PowerStation 4.0. В качестве тестовой задачи выбрана задача о течении Пуазейля [2]. Тесты проводились для 1000 шагов по времени.

На рис. 2 показаны графики зависимости времени расчета задачи от числа частиц. Из приведенных графиков видно, что алгоритм сетки для данной задачи дает ускорение примерно в 60 раз. Оценим его теоретическую сложность.

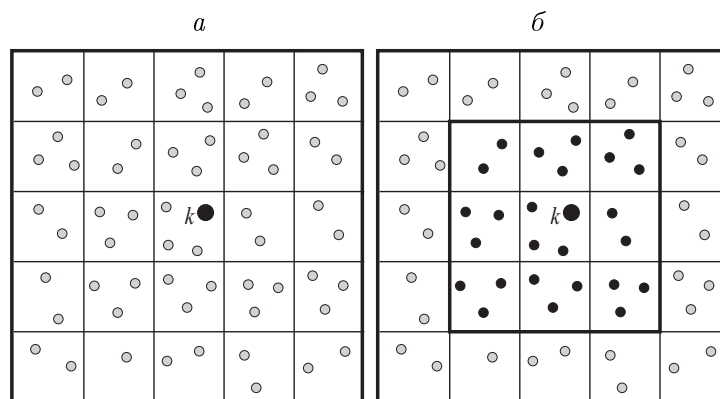


Рис. 1. Сетка (*а*) и ячейки перебора (*б*)

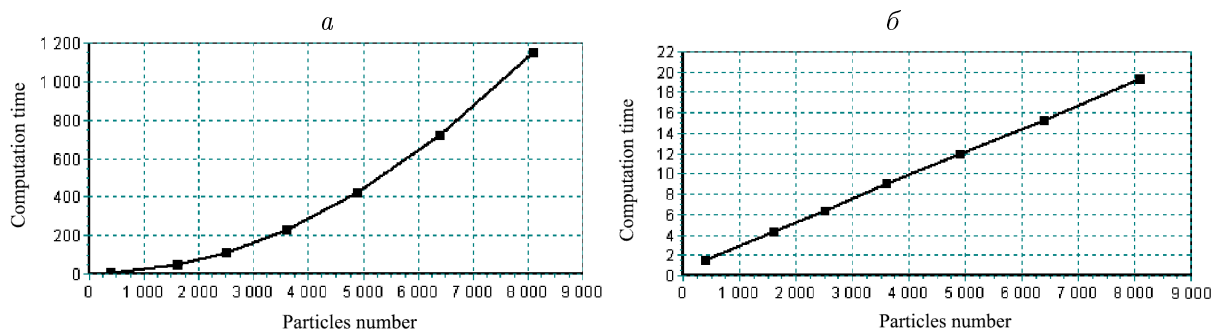


Рис. 2. Зависимость времени расчета задачи от числа частиц: (*а*) — метод полного перебора частиц; (*б*) — с использованием алгоритма сетки

Как правило, для оценки скорости работы алгоритмов применяется так называемая O -символика или O -сложность алгоритмов. O -функции выражают относительную скорость алгоритма в зависимости от некоторой переменной или набора переменных.

Существуют четыре важных свойства для определения сложности алгоритма:

$$O(kf) = O(f), \quad O(fg) = O(f)O(g),$$

$$O(f/g) = O(f)/O(g), \quad O(f+g) \text{ равна доминанте } O(f) \text{ и } O(g).$$

Здесь k — некоторая константа, а f и g — аргументы. В условиях данной задачи в качестве аргумента будет использоваться заданное число частиц. Соответственно, O -функцией будет являться число операций (время работы процедуры поиска).

Пользуясь приведенными свойствами для оценки алгоритмов, получаем следующую оценку скорости метода:

$$O(N) + O(9Np) = O(Np),$$

где N — число частиц области; p — число частиц, находящихся в ячейке сетки.

Далее необходимо определить зависимость вида $p = p(N)$ для записи окончательного выражения для оценки скорости метода сетки. Для упрощения выкладок за $p(N)$ возьмем среднее число частиц в ячейках сетки. Затем выведем зависимость $p = p(N)$ с учетом этого упрощения:

$$h = dx = \frac{L}{m},$$

где h — сглаживающая длина; L — линейный размер области; m — начальное число частиц по одному измерению ($N = mt$). Как было сказано ранее, линейный размер ячейки сетки выбирается следующим образом:

$$l = 2h(1 + \varepsilon) = \frac{2L}{m}(1 + \varepsilon).$$

Тогда число ячеек вдоль линейного размера области можно определить так:

$$M = \frac{L}{l} = \frac{Lm}{2L(1 + \varepsilon)} = O(m),$$

где M — число ячеек по одному измерению (MM — число ячеек сетки в случае квадратной области). Соответственно, для полного числа частиц в области можно записать следующее приближенное выражение:

$$N \approx M^2 p(N). \quad (1)$$

Таким образом, число частиц в области равно произведению числа ячеек области на среднее число частиц в ячейке (используется приближенное равенство, так как усредняется число частиц в каждой ячейке). Выражение (1) можно переписать в виде

$$N = O(m^2)p(N).$$

Так как $N = mt$, выражение для функции $p(N)$ примет вид

$$p(N) = \text{const}. \quad (2)$$

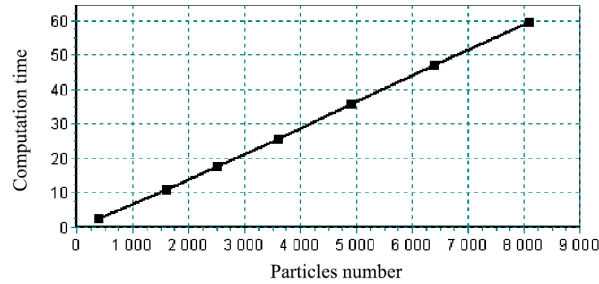


Рис. 3. Ускорение алгоритма сетки

Соотношение (2) означает, что при любом числе частиц разбиение сетки строится таким образом, что число частиц в ячейке всегда примерно одно и то же. В результате функция сложности алгоритма имеет вид

$$O(N). \quad (3)$$

Выражение (3) определяет теоретическую оценку скорости алгоритма сетки исходя из аналитических формул построения сеточного разбиения. Как можно заметить, зависимость имеет линейный характер, что согласуется с полученными численными данными на рис. 2, б и 3.

3. Параллельная реализация метода

Идея параллельной реализации метода SPH основана на используемом алгоритме поиска ближайших соседей. В работе [3] для поиска соседей используются бинарные деревья, что обуславливает соответствующий способ параллельной реализации. В данной работе используется подход, предложенный в [4].

Первоначально область расчета разбивают на геометрически равные подобласти, число которых равно числу используемых в расчете процессоров. Каждый процессор хранит только те частицы, которые лежат в соответствующей ему подобласти. Далее необходимо сделать следующие допущения: любая частица области за один временной шаг может перейти только в соседнюю ячейку (это может быть достигнуто использованием условия Куранта — Фридрихса — Леви для выбора временного шага); размеры ячеек на всех процессорах одинаковые.

Тогда для каждого шага по времени на каждом процессоре необходимо выполнить следующую последовательность операций.

1. Построение сетки и распределение частиц по ячейкам.
2. Пересылка частиц, попавших в граничные ячейки, на смежные процессоры.
3. Поиск соседей для каждой частицы.
4. Расчет характеристик частиц.
5. Перебор частиц в граничных ячейках и их пересылка на смежные процессоры в случае их выхода за пределы подобласти данного процессора.

Для тестирования выбрана та же задача о течении Пуазейля, особенностью которой является равномерность распределения частиц по области расчета. На рис. 4, а приведено время расчета задачи на разном количестве процессоров, на рис. 4, б представлен график ускорения.

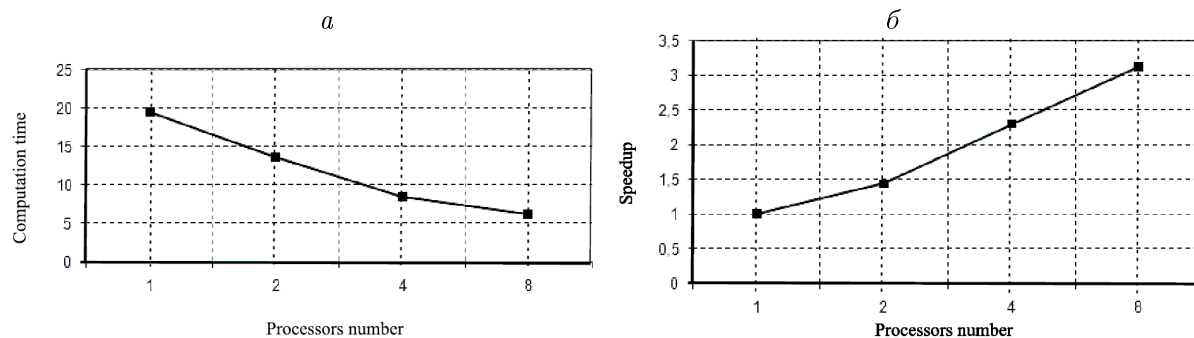


Рис. 4. Время расчета (а) и ускорение (б)

Анализ времени выполнения программы на одном процессоре и на кластере показывает незначительную степень ускорения параллельной реализации. Это связано с тем, что время, затрачиваемое на вычисления, соизмеримо со временем, затрачиваемым на пересылки. Существенным недостатком данной параллельной реализации является также высокая чувствительность к равномерности распределения частиц по расчетной области. Дальнейшая оптимизация данного алгоритма может быть осуществлена с использованием неблокирующих функций пересылок, а также разбиением области на подобласти с приблизительно одинаковым числом частиц. Это позволит не только ускорить время расчета, но и использовать большее количество частиц, что скажется на качестве получаемых результатов.

Заключение

В работе показано, что использование алгоритма сетки по сравнению с полным перебором дает ускорение алгоритма расчета задачи примерно в 60 раз и показывает теоретическую и экспериментальную оценку $O(N)$. Кроме того, ускорение также достигается при “правильном” распараллеливании алгоритма решаемых задач, однако его эффективность будет высокой только при большом количестве расчетных частиц.

Список литературы

- [1] LIU G.R., LIU M.B. Smoothed particle hydrodynamics: a meshfree particle method. World Scientific, 2003.
- [2] АФАНАСЬЕВ К.Е., ИЛЬЯСОВ А.Е., МАКАРЧУК Р.С., ПОПОВ А.Ю. Численное моделирование течений жидкости со свободными границами методами SPH и MPS // Вычисл. технологии. 2006. Т. 11. Спецвыпуск: Избранные доклады семинара по численным методам и информационным технологиям Кемеровского государственного университета. С. 26–44.
- [3] SPRINGEL V., YOSHIDA N., WHITE S.D.M. // GADGET: A code for collisionless and gasdynamical cosmological simulations, arXiv:astro-ph/0003162v3 31 May 2001.
- [4] GOOZEE R.J., JACOBS P.A. Distributed and shared memory parallelism with a smoothed particle hydrodynamics code // ANZIAM J. 2003. Vol. 44 (E). P. 202–228.