РОССИЙСКАЯ АКАДЕМИЯ НАУК
СИБИРСКОЕ ОТДЕЛЕНИЕ
АВТОМЕТРИЯ

2006, том 42, № 3

УДК 539.2, 536.1

ПРИМЕНЕНИЕ НЕЙРОСЕТЕВОГО ПОДХОДА ДЛЯ ПОСТРОЕНИЯ ДВУМЕРНЫХ АДАПТИВНЫХ СЕТОК *

О. И. Нечаева

Институт вычислительной математики и математической геофизики СО РАН, Новосибирск
E-mail: nechaeva@ssd.sscc.ru

Предложен нейросетевой подход к построению конечно-разностных адаптивных сеток, в основе которого лежит алгоритм обучения для самоорганизующихся карт Кохонена. Приведен алгоритм построения сеток на примере двумерных областей на плоскости. Показано, что предлагаемый подход допускает распараллеливание построения сетки с эффективностью более 90 %.

Введение. Для численного решения некоторых сложных прикладных задач требуется построение адаптивных сеток. Использование таких сеток может повысить точность приближенного решения задачи без существенного увеличения числа узлов [1]. Важным требованием к методу построения адаптивных сеток является возможность его эффективного распараллеливания. При решении задачи на адаптивной сетке часто возникает необходимость перестраивать сетку в течение итерационного процесса. Поэтому метод построения должен эффективно распараллеливаться с учетом требований параллельной реализации задачи.

Для решения конечно-разностных задач часто используют адаптивные сетки, которые изоморфны равномерной прямоугольной сетке. Построение таких сеток в общем случае эквивалентно нахождению отображения некоторой вычислительной области на физическую, переводящего равномерную сетку в адаптивную с заданным распределением плотности. В традиционных методах, к которым относятся метод эквираспределения [2], метод Томпсона [3], эллиптический метод [4], такое отображение строится путем решения нелинейных дифференциальных уравнений с частными производными. Поэтому распараллеливание этих методов зачастую сталкивается с непреодолимыми трудностями.

В данной работе предлагается нейросетевой подход для построения конечно-разностных адаптивных сеток, в котором отображение вычислитель-

* Работа выполнена при поддержке Президиума РАН «Программа фундаментальных исследований № 14.15» (2006) и программы Рособразования «Развитие научного потенциала ВШ» (проект РНП.2.2.1.1.3653).

ной области на физическую строится с помощью самоорганизующихся карт Кохонена (Self Organizing Map (SOM)) – нейронной сети [5] (впервые предложенной финским ученым Т. Кохоненом в 1984 году). Областями традиционного применения SOM являются кластеризация и визуализация данных [6], так как SOM позволяет проектировать многомерные данные на пространства меньшей размерности с сохранением топологии. Эта особенность и позволяет применить SOM для построения сеток [7]. К сожалению, в мировой литературе не удалось найти работ о построении конечно-разностных сеток с использованием SOM. Только в [8] SOM применяется при построении адаптивных сеток, предназначенных для метода конечных элементов.

Нейросетевой подход для построения конечно-разностных сеток обладает следующими достоинствами. Во-первых, имеется возможность эффективного распараллеливания построения сетки, которая обеспечивается внутренним параллелизмом нейронной сети SOM [9]. Во-вторых, алгоритм построения сетки в рамках нейросетевого подхода одинаков для любых размерностей пространства и физической области. В-третьих, алгоритм построения прост в реализации и может работать с произвольными начальными данными, например равномерными прямоугольными сетками с узлами внутри области. Построенные адаптивные сетки планируется использовать для моделирования пространственной динамики различных физических, химических и экологических процессов в научных исследованиях.

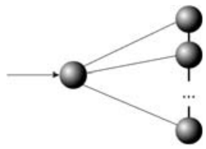
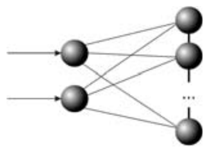
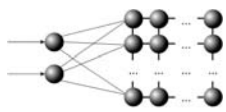
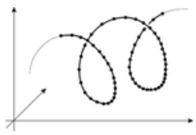
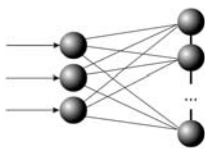
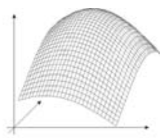
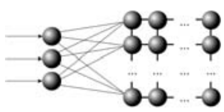
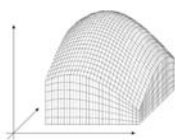
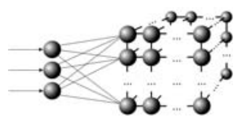
Описание нейросетевого подхода. Нейронная сеть SOM состоит из двух слоев нейронов [5]. Все нейроны первого слоя имеют по одному входу, на которые в процессе обучения подаются координаты случайно сгенерированной точки пространства. Поэтому число нейронов первого слоя равно размерности пространства, в котором находится физическая область. Нейроны второго слоя расположены в узлах решетки, структура которой определяет структуру строящейся адаптивной сетки, а значит, и размерность физической области. Каждый нейрон в этой решетке соединен взвешенными связями со всеми нейронами первого слоя и соответствует узлу сетки, а веса связей играют роль координат соответствующего узла сетки в физической области.

Обучение нейронной сети SOM основано на самоорганизации – «обучении без учителя» [10] – и заключается в настройке весов связей между нейронами первого и второго слоя. На каждой итерации процесса обучения вычисляются расстояния между вектором координат случайной точки пространства, поданных на входы первого слоя, и вектором весов связей для каждого нейрона второго слоя. Нейрон с наименьшим расстоянием объявляется победителем, после чего проводится корректировка весов связей всех нейронов второго слоя, направленная на уменьшение вычисленных расстояний. При этом наибольшее изменение получают веса связей нейрона-победителя, а веса связей остальных нейронов меняются тем меньше, чем дальше они расположены от победителя в решетке.

За распределение плотности адаптивной сетки отвечает вероятностное распределение для генерации очередной случайной точки. В результате обучения веса нейронов второго слоя настраиваются таким образом, что нейроны, близкие друг к другу в решетке, соответствуют близким точкам пространства. Поэтому в процессе обучения изначально равномерная прямоугольная сетка, расположенная внутри области, постепенно растягивается по области, сгущаясь в нужных нам участках. Настройка весов связей нейронов производится по одному и тому же правилу для каждого нейрона, вследствие чего новое положение узла сетки на каждой итерации вычисляется независимо от других узлов. В этом заключается внутренний параллелизм метода построения, и его распараллеливание будет эффективно при любом равномерном распределении узлов сетки по процессорам.

Нейросетевой подход использует один и тот же алгоритм построения адаптивных сеток для любых размерностей пространства и области. Меняя количество нейронов в первом слое и структуру второго слоя нейронной сети SOM, можно получить нужную конфигурацию размерностей. В таблице перечислены варианты конфигураций размерностей и соответствующие нейронные сети.

Конфигурации размерностей пространства и физической области

№ п/п	Пространство	Область	Сетка	Нейронная сеть
1	R^1	Отрезок прямой		
2	R^2	Отрезок кривой на плоскости		
3		Плоская область		
4	R^3	Отрезок пространственной области		
5		Поверхность второго порядка		
6		Тело		

Таким свойством не обладают традиционные методы построения сеток. Для каждой конфигурации размерностей в этих методах выводятся свои нелинейные дифференциальные уравнения, сложность решения которых существенно возрастает с увеличением размерностей задачи. Нейросетевой подход в данной работе рассматривается для двумерных областей на плоскости (см. таблицу, строка № 3).

Алгоритм построения адаптивной сетки для двумерных областей на плоскости. Пусть G – двумерная физическая область в пространстве $R^2 = \{x \mid x = (x^1, x^2)\}$, в которой требуется построить адаптивную сетку размером $N_1 \times N_2$ (рис. 1, *c*). Задача построения конечно-разностной адаптивной сетки на физическую область G в общем виде эквивалентна нахождению отображения ξ вычислительной области, в качестве которой обычно выступает единичный квадрат

$$Q = \{(q^1, q^2): 0 \le q^1 \le 1,\ \ 0 \le q^2 \le 1\},$$

на область G в виде

$$\xi: (q^1, q^2) \to (x^1, x^2), \qquad x^k = x^k(q^1, q^2), \quad k = 1,2, \tag{1}$$

переводящего равномерную сетку

$$Q_N = \{q_{ij} = (q_{ij}^1,\ q_{ij}^2),\ \ i = 0,\dots, N_1 - 1,\ \ j = 0,\dots, N_2 - 1\}$$

(рис. 1, *a*) в адаптивную сетку

$$G_N = \{x_{ij} = (x_{ij}^1, x_{ij}^2),\ i = 0,\dots, N_1 - 1,\ j = 0,\dots, N_2 - 1\}$$

с распределением плотности, заданным неотрицательной управляющей функцией w: $G \to R^+$. При этом достаточно рассматривать ξ как дискретное отображение, определенное только на узлах сетки Q_N.

При нейросетевом подходе в качестве начального приближения искомого отображения ξ можно использовать любое линейное преобразование, переводящее сетку Q_N внутрь области G, т. е. начальная сетка $G_N(0)$ может выглядеть, например, как показано на рис. 1, *b*. Эта сетка будет постепенно растягиваться по области, тем самым приближая искомое ξ.

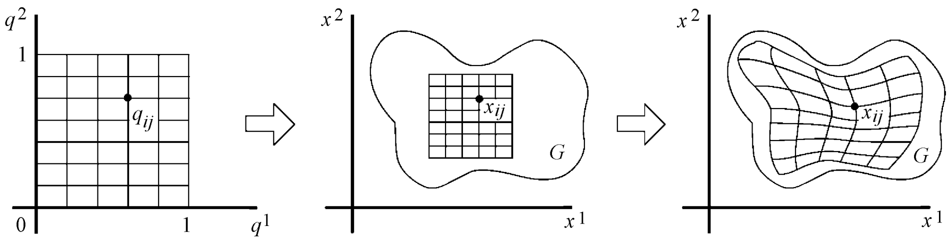


Рис. 1. Построение адаптивной сетки: вычислительная область (*a*), физическая область с начальной сеткой (*b*), физическая область с сеткой в процессе построения (*c*)

Плотность вероятностного распределения для генерации случайных точек, которые будут подаваться на входы первого слоя нейронов, задается нормированной управляющей функцией

$$p(x) = \frac{w(x)}{\int\limits_G w(x)dx}. \quad (2)$$

Алгоритм построения адаптивной сетки, основанный на обучении нейронной сети SOM [10], состоит из следующих шагов.

А л г о р и т м 1.

0. Устанавливаются начальные положения узлов $x_{ij}(0) = (x_{ij}^1(0), x_{ij}^2(0))$, $i = 0, \ldots, N_1 - 1,\ j = 0, \ldots, N_2 - 1.$

1. На каждой итерации с номером t выполняются следующие действия:

а) генерируется случайная точка $y(t) = (y^1(t), y^2(t))$ из области G с учетом заданной вероятностной плотности $p(x)$, вычисляемой по формуле (2);

б) выбирается ближайший к точке $y(t)$ узел $x_{mn}(t)$ по условию

$$\left\| y(t) - x_{mn}(t) \right\| \le \left\| y(t) - x_{ij}(t) \right\| \quad (3)$$

для всех $i = 0, \ldots, N_1 - 1,\ j = 0, \ldots, N_2 - 1,$ который называется победителем;

в) проводится корректировка положений узлов сетки в соответствии с формулой

$$x_{ij}(t+1) = x_{ij}(t) + \theta_{mn}(t, i, j)(y(t) - x_{ij}(t)) \quad (4)$$

для всех $i = 0, \ldots, N_1 - 1,\ j = 0, \ldots, N_2 - 1,$ где $\theta_{mn}(t, i, j) \in (0,1]$ – коэффициент обучения.

2. Повторяется шаг 1, пока изменения положений узлов сетки не станут достаточно малыми.

На каждой итерации алгоритма узлы сетки сдвигаются к сгенерированной точке $y(t)$. Величина сдвига для каждого узла определяется коэффициентом обучения $\theta_{mn}(t, i, j)$. От выбора этого коэффициента зависят качество и скорость построения сетки.

Выбор коэффициента обучения. В работах [10, 11] общий вид коэффициента обучения – это произведение шага и радиуса обучения:

$$\theta_{mn}(t, i, j) = \delta(t)\,\eta_{mn}(t, i, j). \quad (5)$$

Шаг обучения $\delta(t)$ – это убывающая с числом итераций функция, которая должна обеспечивать остановку итерационного процесса за счет уменьшения величины сдвигов узлов сетки. При подборе шага обучения нужно учитывать, что при быстром убывании $\delta(t)$ сетка может не успеть растянуться по области до остановки алгоритма, а при медленном убывании $\delta(t)$ время работы алгоритма может увеличиться без существенного улучшения качества построения сетки. Проведен ряд экспериментов, на основе которых подобрана функция $\delta(t) = t^{0,2}$. Для большинства тестовых областей $\delta(t)$ давала сетки лучшего качества [7].

Радиус обучения $\eta_{mn}(t, i, j)$ отвечает за распределение величин сдвигов по узлам сетки на итерации t. Он должен обеспечивать условия, при которых наибольший сдвиг получает узел-победитель $x_{mn}(t)$, а остальные узлы $x_{ij}(t)$ сдвигаются тем меньше, чем дальше они расположены от победителя в решетке нейронов. Чтобы значения коэффициента обучения $\theta_{mn}(t,i,j)$ находились в пределах (0, 1] для каждого t, должно выполняться условие $|\eta_{mn}(t,i,j)| \le 1$. Поэтому радиус обучения в [10] выбирается в виде экспоненты:

$$\eta_{mn}(t,i,j) = e^{-\frac{(i-m)^2+(j-n)^2}{2\sigma^2(t)}}, \tag{6}$$

которая для каждого t принимает максимальное значение $\max\limits_{(i,\,j)}(\eta_{mn}(t,i,j)) = 1$ на индексах узла-победителя (n,m) и убывает во всех направлениях от него в решетке нейронов. Проведен ряд экспериментов, на основе которых была подобрана функция $\sigma(t) = at^{0,2}$, стоящая в знаменателе показателя степени экспоненты (6). Для большинства тестовых областей $\sigma(t)$ давала сетки лучшего качества [7].

Константа a должна обеспечивать условия, при которых на первой итерации величина сдвига каждого узла $x_{ij}(0)$ не меньше βd, где β – заданное перед началом построения число, $0<\beta<1$, а d – расстояние от x_{ij} до сгенерированной точки $y(0)$. Значение β для приведенных в данной работе примеров выбиралось в пределах $0{,}01<\beta<0{,}10$. Формула для вычисления значения a получается следующим образом. Нужно взять два узла, наиболее удаленные друг от друга в решетке нейронов, например x_{00} и $x_{N_1-1,\,N_2-1}$. Назначив x_{00} победителем, можно заметить, что минимальное значение функции $\eta_{mn}(t,i,j)$ будет достигаться на узле $x_{N_1-1,\,N_2-1}$. Потребовав, чтобы это значение при $t=1$ было равно β, можно получить уравнение для нахождения значения a:

$$a = \sqrt{-\frac{(N_1-1)^2+(N_2-1)^2}{2\ln\beta}}. \tag{7}$$

Понятно, что при любом другом победителе, отличном от x_{00}, будет выполняться $\min\limits_{(i,\,j)}(\eta_{mn}(1,i,j)) \ge \beta$. На рис. 2 на каждом графике в нижней плоскости расположена решетка нейронов, а по оси θ отмечается величина сдвигов соответствующих узлов сетки в области G.

Итерационный процесс построения сетки останавливается, когда изменения положений узлов становятся достаточно малыми. Так как максимальный сдвиг получает победитель, то при проверке условия остановки достаточно учитывать только этот сдвиг. Поскольку $\theta_{mn}(t,i,j)\big|_{\substack{i=m\\ j=n}} = \delta(t)$, корректировку положения победителя $x_{mn}(t)$ можно записать в виде

$$x_{mn}(t+1) = x_{mn}(t) + \delta(t)(y - x_{mn}(t)).$$

Перед началом работы алгоритма задается малое ε, $0\le\varepsilon\le1$, и остановка происходит после того, как выполнилось условие $\delta(t)\le\varepsilon$. При этом заранее может быть вычислено количество итераций до остановки с помощью цело-

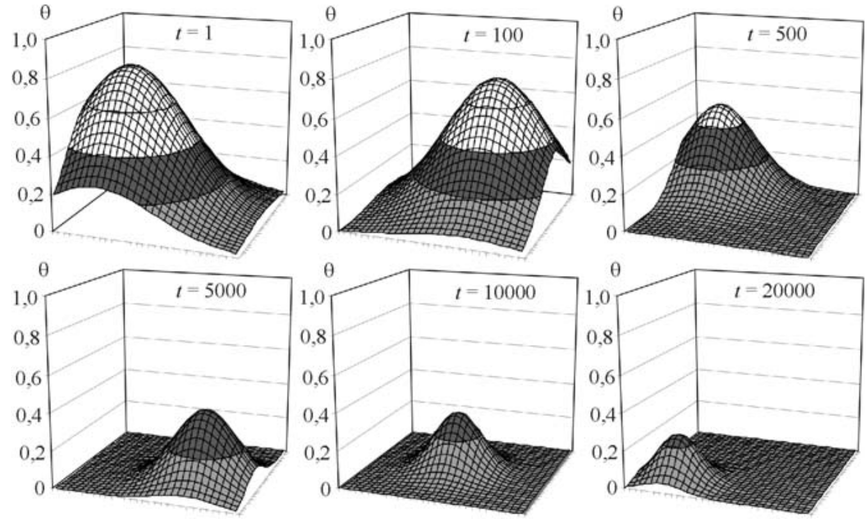


Рис. 2. Результирующие сдвиги узлов на различных итерациях

численного решения уравнения $\varepsilon = \delta(t)$. Если $\delta(t) = t^{0,2}$, то это количество равно $\lceil \varepsilon^{-5} \rceil$.

Модификация алгоритма построения адаптивных сеток. Если область G выпуклая, то узлы сетки в процессе ее построения перемещаются внутри области G, не выходя за границу. Действительно, выпуклость области означает, что для любой пары точек x_1 и x_2 из G все точки отрезка $[x_1, x_2]$ принадлежат области G. Поэтому если узел $x_{ij}(t)$ лежит в области G, то следующее его положение – точка $x_{ij}(t+1)$ – также будет принадлежать G, так как $y \in G$ и $x_{ij}(t+1) \in [x_{ij}(t), y] \subset G$. На практике, даже если на нулевом шаге узлы сетки лежат за пределами G, в процессе работы алгоритма они втянутся внутрь области. Однако для построения сеток на произвольных областях непосредственное применение описанного выше алгоритма не всегда дает приемлемые результаты. Возникают следующие проблемы.

1. Если область G не является выпуклой, то не гарантируется, что все узлы сетки G_N будут принадлежать G.

2. Граничные узлы построенных сеток расположены на определенном расстоянии до границы области, отличном от нуля. Это расстояние сопоставимо со средним расстоянием между узлами сетки.

Для решения проблем предлагается следующая модификация алгоритма построения. В работе [8], которая посвящена использованию SOM для построения конечно-элементных сеток, была предложена идея модификации алгоритма обучения SOM, состоящая в чередовании применения этого алгоритма для граничных и внутренних узлов. Один цикл такого чередования называется далее макроитерацией. В соответствии с этой идеей разработан модифицированный алгоритм построения конечно-разностных адаптивных сеток.

А л г о р и т м 2.

0. Инициализация положений узлов сетки.

1. Применение алгоритма 1 в течение n_0 итераций ко всем узлам сетки.

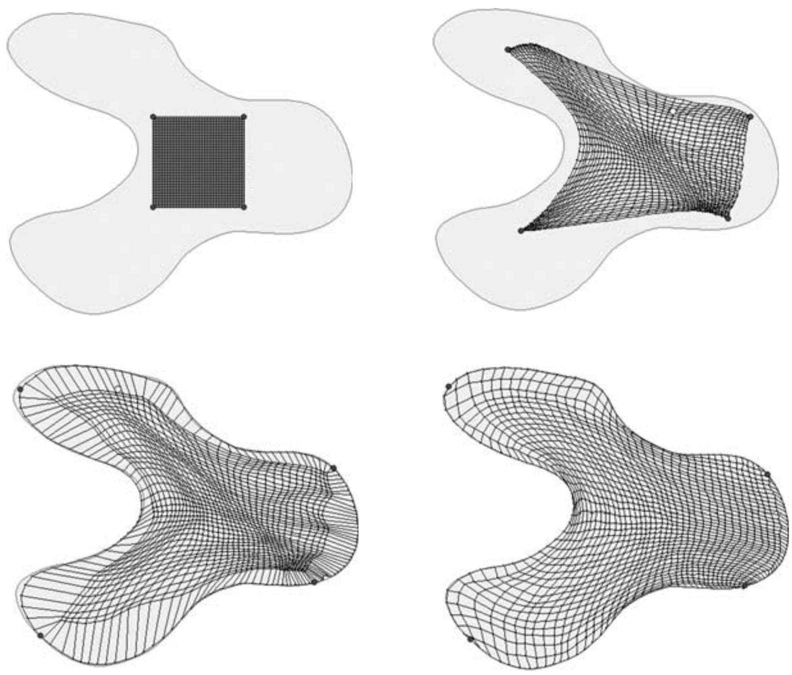


Рис. 3. Процесс построения сетки с помощью алгоритма 2

2. На каждой макроитерации с номером s выполняются следующие действия:

а) применение алгоритма 1 в течение $n_1(s)$ итераций к граничным узлам сетки с генерацией точки только на границе области;

б) применение алгоритма 1 в течение $n_2(s)$ итераций ко всем узлам с генерацией точки во всей области, при этом все граничные узлы зафиксированы и не меняют своего положения.

3. Повторяются макроитерации до тех пор, пока изменения положений узлов не станут достаточно малыми.

После первого шага модифицированного алгоритма сдвиги узлов еще остаются достаточно большими, но сетка уже грубо приняла форму области G, а граничные узлы расположились вблизи той части границы, на которой они будут стоять в построенной сетке. За счет того что на шаге 2б алгоритм 1 применяется ко всем узлам, а не только к внутренним, остается согласованность между граничными и внутренними узлами и связь между ними не нарушается. Также участие граничных узлов в выборе победителя приводит к тому, что внутренние узлы быстрее подтягиваются к границе области. На шаге 2а целесообразно использовать меньшие значения константы a в коэффициенте обучения, так как фактически строится одномерная сетка на кривой (см. таблицу, строка № 2). На рис. 3 показано построение сетки модифицированным алгоритмом.

Параллельный алгоритм построения сетки. Внутренний параллелизм нейронной сети SOM, которая лежит в основе нейросетевого подхода, дает возможность эффективного распараллеливания описанного алгоритма построения адаптивных сеток [9]. Наиболее трудоемкими в последовательном алгоритме 1 являются операции выбора узла-победителя (3) и кор-

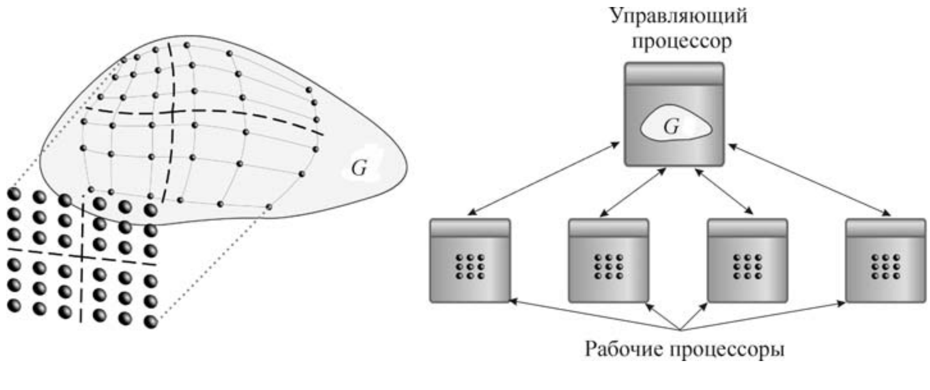


Рис. 4. Распараллеливание алгоритма построения сеток при нейросетевом подходе

ректировки положений узлов (4), так как они требуют перебора почти всех узлов сетки. Однако при нахождении победителя расстояния $\| y(t) - x_{ij}(t) \|$, $i = 0, \ldots, N_1 - 1$, $j = 0, \ldots, N_2 - 1$, от узлов сетки до сгенерированной точки можно вычислять независимо для каждого узла, а при корректировке сдвиг каждого узла определяется по формуле (4) независимо от других узлов сетки. Поэтому эти операции можно выполнять параллельно.

Идея распараллеливания алгоритма построения сетки (рис. 4) заключается в том, что массив узлов разрезается и распределяется по процессорам. Удобно также выделить управляющий процессор, который будет хранить информацию о входной области и согласовывать действия рабочих процессоров.

В течение работы параллельного алгоритма управляющий процессор генерирует случайную точку из области и рассылает ее координаты всем рабочим, каждый из которых после этого находит локальный узел-победитель и отсылает его индексы управляющему. Индексы глобального победителя и номер процессора, в котором он находится, вычисляются в управляющем процессоре, пересылающем эту информацию рабочим для корректировки положений узлов сетки.

Важнейшей особенностью параллельного алгоритма является то, что коммуникации происходят только между управляющим и рабочими процессорами. При этом в течение одной итерации пересылается небольшое количество информации, например в двумерном случае всего 32 байта на каждый рабочий процессор независимо от размера массива узлов. Вследствие этого эффективность распараллеливания составляет более 90 %, и если каждый рабочий процессор хранит примерно одинаковое количество узлов, то эффективность не зависит от способа разрезания массива. Поэтому распределять этот массив можно в соответствии с требованиями задачи, решаемой на сетке, а значит, обеспечивается возможность эффективно подстраивать сетку в процессе решения задачи в том же параллельном режиме, без перераспределения узлов сетки между процессорами и независимо от расположения узлов в области *G*. Параллельная программа для алгоритма построения адаптивных сеток была реализована с использованием библиотеки MPI и тестировалась на кластере Сибирского суперкомпьютерного центра СО РАН.

Заключение. В работе предложен нейросетевой подход для построения конечно-разностных адаптивных сеток. Способность нейронной сети SOM, лежащей в основе этого подхода, отображать многомерные данные на пространства меньшей размерности позволяет использовать один и тот же алгоритм построения сетки для различных размерностей пространства и физической области. Нейросетевой подход продемонстрирован для двумерных областей на плоскости. Приведен алгоритм построения адаптивных сеток и его модификация, которая позволяет решить проблемы, связанные с выходом узлов сетки за границу невыпуклых областей и неплотным прилеганием сетки к границе.

Показано, что эффективность распараллеливания алгоритма построения сетки составляет более 90 %. При этом распределение узлов сетки по процессорам может выполняться в соответствии с требованиями на параллельную реализацию задачи, решаемой на сетке. В дальнейшем планируется провести сравнительный анализ нейросетевого подхода с традиционными методами построения адаптивных сеток для различных размерностей пространства и физической области.

СПИСОК ЛИТЕРАТУРЫ

1. **Лебедев А. С., Лисейкин В. Д., Хакимзянов Г. С.** Разработка методов построения адаптивных сеток // Вычисл. технологии. 2002. **7**, № 3. С. 29.
2. **Хакимзянов Г. С., Шокин Ю. И., Барахнин В. Б., Шокина Н. Ю.** Численное моделирование течений жидкости с поверхностными волнами. Новосибирск: Изд-во СО РАН, 2001.
3. **Thompson J. F., Warsi Z. U. A., Mastin C. W.** Numerical Grid Generation, Foundations and Applications. Amsterdam: North-Holland, 1985.
4. **Лисейкин В. Д., Лебедев А. С., Китаева И. А.** Универсальный эллиптический метод построения разностных сеток. Новосибирск: Изд-во НГУ, 2004.
5. **Kohonen T. K.** Self-Organization and Associative Memory. N. Y.: Springer-Verlag, 1989.
6. **Flexer A.** On the use of self-organizing maps for clustering and visualization // Intelligent Data Analysis. 2001. **5**. P. 373.
7. **Нечаева О. И.** Применение самоорганизующихся карт Кохонена (SOM) для построения адаптивных сеток // Тр. конф. молодых ученых ИВМиМГ. Новосибирск: ИВМиМГ СО РАН, 2004. С. 140.
8. **Manevitz L., Yousef M., Givoli D.** Finite-element mesh generation using self-organizing neural networks // Microcomputers in Civil Eng. 1997. **12**. P. 233.
9. **Nechaeva O.** Neural network approach for parallel construction of adaptive meshes // Lecture Notes in Comput. Sci. /Ed. V. Malyshkin. Berlin: Springer, 2005. Vol. 3606. P. 446.
10. **Ritter H., Martinetz T., Schulten K.** Neural Computation and Self-Organizing Maps: An Introduction. N. Y: Addison-Wesley, 1992.
11. **Ghahramani Z.** Unsupervised learning // Lecture Notes in Artificial Intelligence /Eds. O. Bousquet et al. Berlin–Heidelberg: Springer-Verlag, 2004. Vol. 3176. P. 72.

Поступила в редакцию 3 ноября 2005 г.