Распараллеливание алгоритма обучения радиально-базисной нейронной сети для решения краевой задачи математической физики на графическом процессоре в технологии CUDA

Н. О. Матвеева

Одной из важных проблем, возникающих при решении задач математической физики, является большие вычислительные затраты. Но многим методам решения задач математической физики присущ внутренний параллелизм вычислений, что позволяет использовать параллельные системы для их решения. Одним из наиболее эффективных бессеточных методов решения дифференциальных уравнений в частных производных является построение и обучение радиально-базисной нейронной сети (RBFNN). RBFNN могут быть реализованы на параллельных системах. На сегодняшний день для ускорения вычислений активно используются графические процессоры с массивно-параллельной архитектурой.

Радиально-базисная нейронная сеть представляет собой сеть с двумя слоями. Первый слой осуществляет преобразование входного вектора $\mathbf{x}$ с использованием радиально-базисных функций (RBF). Выход сети описывается выражением $u = \sum_{k=1}^{m} w_k \varphi_k(\mathbf{x})$, где $\varphi_k(\mathbf{x})$ – RBF, w_k – вес, связывающий выходной нейрон с k-ым нейроном первого слоя; m – число нейронов первого слоя.

Для обучения сети использовался градиентный алгоритм обучения. Градиентный алгоритм спуска, одновременно оптимизирующий веса, центры и ширину может быть построен в виде последовательности двух шагов. Шаг 1. Зафиксировав центры и ширину, находим веса, минимизирующие функционал ошибки. Для минимизации весов сети использовался алгоритм сопряженных градиентов. Шаг 2. Зафиксировав веса, находим центры и ширину.

Алгоритм обучения радиально-базисной нейронной сети был реализован с помощью технологии CUDA и Visual Studio 2005 на графическом процессоре nVidia GeForce 8800 GTX, на центральном процессоре Intel Pentium 4 – в системе MATLAB 7.7.0. Экспериментальное исследование проводилось на примере модельной задачи решения двумерного уравнения Пуассона. Анализ результатов экспериментов показал, что часть алгоритма не включающая минимизацию весов сети распараллеливается очень эффективно, причем, если для центрального процессора эти вычисления занимают большую часть времени, то для графического большую часть времени занимает коррекция весов методом сопряженных градиентов. Это связано с относительно небольшим числом нейронов.

Таблица. 1. Результаты экспериментов

Параметры сети	Время вычислений для одной итерации, с		Ускорение
	GPU	CPU	
$m = 64$ Число контр. точек 224	0,0035	0,0636	18
$m = 128$ Число контр. точек 224	0,0041	0,1286	31,3
$m = 64$ Число контр. точек 524	0,00387	0,1867	48
$m = 128$ Число контр. точек 524	0,0048	0,3553	74

В таблице 1 приведены результаты экспериментов, из которых видно, что уже при небольшом числе нейронов и контрольных точек реализация алгоритма обучения на графическом процессоре дает существенный выигрыш во времени. При увеличении числа контрольных точек и нейронов было достигнуто ускорение в 74 раза.