УДК 004.032.26

Н.Г. Аксак, А.Ю. Тыхун

*Харьковский национальный университет радиоэлектроники, Харьков*

# АНАЛИЗ МОДЕЛЕЙ ПРОИЗВОДИТЕЛЬНОСТИ РЕАЛИЗАЦИИ НЕЙРОАЛГОРИТМА

*Проведен анализ моделей производительности для декомпозиции нейроалгоритма на уровнях реализации функции и скалярных операций без учета логической топологии коммуникационных связей вычислительной системы.*

***интеллектуальная обработка, нейрокомпьютер, многопроцессорная система, параллельный алгоритм, декомпозиция, градиентный метод, скалярные операции, производительность***

## Введение

С развитием программного и аппаратного обеспечения в разнообразных областях растет необходимость интеллектуальной обработки больших объемов данных для решения различных задач, требующих значительных вычислительных ресурсов. Для решения таких задач хорошо зарекомендовали себя нейронные сети. При этом для естественного способа реализации нейронных сетей подходят нейрокомпьютеры.

Нейрокомпьютеры представляют собой вычислительную систему, архитектура которой ориентирована на выполнение операций, заданных структурой нейронной сети. Нейрокомпьютеры являются вычислителями нового класса, обеспечивающими более быстрое, более дешевое и качественное решение конкретных задач. Основным достоинством использования нейрокомпьютеров является их сверхвысокое быстродействие, толерантность к ошибкам, способность к обучению, способность к обработке данных при условиях сильных помех и искажений. Однако на сегодняшний день вызывает сложность как разработка элементной базы нейрокомпьютеров, так и применение нейросетевых вычислительных технологий.

Прямой перенос имеющихся параллельных алгоритмов и кодов на различные многопроцессорные системы часто неэффективен, поскольку разработка алгоритмов осуществляется без учета логических и физических коммуникационных связей архитектуры системы. В результате появилась необходимость в отображении сложных нейрозадач в условиях ограниченных вычислительных и временных ресурсов, возрастающем объеме вычислительных данных. Таким образом, при решении нейрозадач можно применить распределенные вычисления, учитывая архитектуры многопроцессорных вычислительных систем и параллельные методы, диктуемые спецификой обрабатываемых данных нейросетей.

В настоящее время существуют различные методы декомпозиции алгоритмов [1]: естественный способ параллельной декомпозиции, декомпозиция по ансамблю, декомпозиция на основе принципа перемешивания, декомпозиция по индексирующей переменной и др.

Естественный способ параллельной декомпозиции является наиболее универсальным методом, который позволяет сформулировать параллельный алгоритм и оптимальным образом отобразить на вычислительную архитектуру, эксплуатируя внутреннюю специфику методов и данных решаемой задачи. В рамках естественной парадигмы классический процесс проектирования параллельного алгоритма сводится к следующим этапам:

1. Параллельная декомпозиция задачи на заданном уровне реализации алгоритма.
2. Построение возможных коммуникационных графов и их аналитическое описание в форме модели параллельной производительности.
3. Реализация принципа конкуренции параллельных алгоритмов в целях отображения на многопроцессорную вычислительную архитектуру наилучшего алгоритма.

Применение данного способа в общем случае приводит к конечному набору возможных параллельных декомпозиций исходного алгоритма и выбору наилучшей комбинации способов распараллеливания, приводящих к наибольшему ускорению на вычислительной архитектуре с заданными параметрами.

**Постановка задачи.** На однопроцессорной системе обучение многослойного персептрона (МП) с архитектурой $N_i - N_j - N_k$ с помощью обратного распространения ошибки осуществляется последовательно на уровне скалярных операций [2], где $N_i, N_j, N_k$ – количество нейронов соответственно во входном, скрытом и выходном слоях, $t$ – количество итераций. Общее количество скалярных операций алгоритма обучения МП определяется в соответствии с выражениями:

– вычисление выходов нейронов скрытого слоя $N_Y^{(1)} = 2tN_iN_j$;

 

– вычисление ошибки скрытого слоя $N_E^{(1)} = tN_j\left[3N_k + 7\right]$;

– настройка весовых коэффициентов $w_{ij}^{(1)}$ скрытого слоя $N_W^{(1)} = tN_iN_j\left[3N_k + 8\right]$;

– вычисление выходов нейронов выходного слоя $N_Y^{(2)} = 2tN_jN_k$ (для упрощения изложения материала будем считать вычисление функций активации одной скалярной операцией);

– вычисление ошибки выходного слоя $N_E^{(2)} = 8tN_k$;

– настройка весовых коэффициентов $w_{jk}^{(2)}$ выходного слоя $N_W^{(2)} = 8tN_jN_k$.

Таким образом, для обучения трехслойного персептрона требуется $N_{skal.op.} = t\cdot[7N_j + 8N_k + 10N_iN_j + 13N_jN_k + 3N_iN_jN_k]$ операций. При этом будем понимать, что одна операция выполняется за один такт времени, т. е. обучение МП на последовательном компьютере займет $N_{skal.op.}$ тактов.

Существующие программные реализации нейронных сетей не достаточно полно реализуют ресурсы многопроцессорных вычислительных систем. Самым практичным способом достижения параллельного выполнения является нейроалгоритм с использованием естественного способа декомпозиции на уровне реализации функций нейронов (например, с использованием библиотеки MPI, реализующей механизм передачи сообщений над стандартными сетевыми протоколами).

В данной работе предлагаются естественный способ декомпозиции на уровне скалярных операций для распараллеливания алгоритма обучения МП на архитектуре с массовым параллелизмом и две модели производительности.

## Аналитическое описание производительности

Предлагаемые декомпозиции алгоритма обучения МП с помощью обратного распространения ошибки основаны на выборе и обработке независимых элементов, каждый из которых обрабатывается на отдельном вычислительном узле. Можно выделить три основных фактора взаимозависимости параметров архитектуры системы и уровня реализации алгоритма, логической топологии, количества независимых скалярных операций:

– количество независимых скалярных операций варьируется в зависимости от алгоритмической постановки и независимо от способа реализации алгоритма;

– количество требуемых процессоров и временных тактов варьируется в зависимости от способа реализации алгоритма;

– производительность коммуникационной среды зависит от параметров логической и физической топологии.

Модель производительности параллельного алгоритма основана на двух параметрах:

1. ускорение $S_p = T_1/T_p$, где $T_1, T_p$ – время выполнения параллельного алгоритма на одном и p процессорах соответственно;
2. эффективность $\varepsilon = S_p/p$, определяющая реальную выгоду от использования многопроцессорных систем по сравнению с однопроцессорными.

**Параллелизм алгоритма на уровне реализации функций нейронов.** Определение количества независимых скалярных операций при параллелизме алгоритма на уровне реализации функций нейронов осуществляется следующим образом: в качестве независимых операций, выполняющихся параллельно, представлены функции вычисления выхода нейрона, ошибки и настройки весовых коэффициентов, а скалярные операции для вычисления перечисленных функций выполняются последовательно на отдельных вычислительных узлах. Определение количества процессоров и тактов, необходимых для обработки алгоритма обучения, рассмотрим на примере настройки весовых коэффициентов скрытого слоя $w_{ij}^{(1)}$.

При настройке весовых коэффициентов

$$w_{ij}^{(1)T} = \begin{bmatrix} w_{01}^{(1)} w_{11}^{(1)} \ldots w_{i1}^{(1)} \\ w_{02}^{(1)} w_{12}^{(1)} \ldots w_{i2}^{(1)} \\ \ldots \\ w_{0j}^{(1)} w_{1j}^{(1)} \ldots w_{ij}^{(1)} \end{bmatrix}$$

скрытого слоя вычислительная нагрузка обработки независимых операций, выполняющихся параллельно на всех вычислительных узлах, распределяется на $N_{max} = \overline{1, N_j}$ процессорах и требует $N_i + 1$ тактов:

на первом такте вычисляется вектор

$$W_0 = [w_{01}^{(1)}, w_{02}^{(1)}, w_{03}^{(1)}, \cdots, w_{0j}^{(1)}],$$

где $w_{01}^{(1)} = b_1^{(1)}, \ldots, w_{0j}^{(1)} = b_j^{(1)}$,

на 2-м такте вычисляется вектор

$$W_1 = [w_{11}^{(1)}, w_{12}^{(1)}, w_{13}^{(1)}, \cdots, w_{1j}^{(1)}],$$

на 3-м такте вычисляется вектор

$$W_2 = [w_{21}^{(1)}, w_{22}^{(1)}, w_{23}^{(1)}, \cdots, w_{2j}^{(1)}],$$

на $N_i + 1$-м такте – $W_i = [w_{i1}^{(1)}, w_{i2}^{(1)}, w_{i3}^{(1)}, \cdots, w_{ij}^{(1)}]$.

Каждый вычислительный узел отдельно осуществляет последовательную обработку $4N_k + 8$ скалярных операций. Таким образом, настройка весовых коэффициентов $w_{ij}^{(1)}$ скрытого слоя требует $N_j$ процессоров и $(4N_k + 8)(N_i + 1)$ тактов.

Обозначим через L количество нейронов в самом широком слое МП. Тогда для реализации алгоритма обучения максимальное количество необходимых процессоров соответствует величине L, а количество временных тактов, необходимых для одной эпохи обучения, равно

$$T_{max} = 2N_i + 2N_j + 6N_k + 31.$$

**Параллелизм алгоритма на уровне реализации скалярных операций.** Определение количества независимых скалярных операций при параллелизме алгоритма на уровне реализации скалярных операций осуществляется посредством выделения тактов обработки независимых скалярных операций, выполняющихся параллельно на всех вычислительных узлах.

Коррекция весовых коэффициентов скрытого слоя определяется в соответствии

$$w_{ij}^{(1)}(t+1) = w_{ij}^{(1)}(t) + 2\alpha^2\gamma_{ij}y_j^{(1)}(1-y_j^{(1)})x_i\sum_{h=1}^{k}\vartheta_{lk}^{(I)}.$$

Определение количества требуемых процессоров и тактов рассмотрим на примере настройки весового коэффициента $w_{11}^{(1)}$. Распределим вычисления скалярных операций на три шага, причем I-й и III-й шаги могут выполняться параллельно, а I-й и II-й шаги взаимозависимы:

I. Определим количество необходимых процессоров и тактов для вычисления $N_k$ слагаемых $\vartheta_{lk}^{(I)} = y_k^{(2)}(1-y_k^{(2)})w_{lk}^{(2)}$.

Опираясь на предположение о независимости элементов, выделим скалярные операции $C_p^{(l)}$: независимые $C_1^{(1)} = w_{lk}^{(2)}y_k^{(2)}$, $C_2^{(1)} = (1-y_k^{(2)})$ и взаимозависимые $C_1^{(2)} = C_1^{(1)}C_2^{(1)}$, где $l$ – номер такта, p – номер процессора.

На этом шаге максимальное количество необходимых процессоров соответствует $P_{max}^{(I)} = 2N_k$, а количество тактов $T_{max}^{(I)} = 3$.

II. При вычислении суммы $\vartheta_{lk}^{(II)} = \sum_{h=1}^{k}\vartheta_{lk}^{(I)}$ для достижения максимального ускорения алгоритма обучения используем алгоритм сдваивания. На рис. 1 представлен алгоритм сдваивания при количестве нейронов скрытого слоя $N_k = 2^m$, количество необходимых временных тактов для осуществления алгоритма сдваивания соответствует $m = \log_2 N_k$.

На этом шаге максимальное количество необходимых процессоров $P_{max}^{(II)} = \frac{N_k}{2}$, $m = T_{max}^{(II)}$ – количество необходимых тактов. Параллельная схема вычисления суммы $\vartheta_{jk}^{(II)}$ представлена на рис. 2.

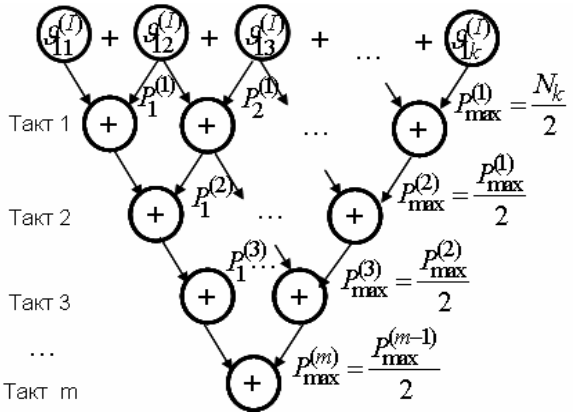


Рис. 1. Алгоритм сдваивания

III. При вычислении $\vartheta_{11}^{(III)} = w_{11}^{(1)}(t) + 2\alpha^2\gamma_{11}y_1^{(1)}(1-y_1^{(1)})x_1$ определение количества процессоров и тактов аналогично I-му шагу.

На этом шаге количество необходимых процессоров соответствует $P_{max}^{(III)} = 4$, а количество тактов – $T_{max}^{(III)} = 5$.

Таким образом, для максимального ускорения вычисления весового коэффициента $w_{11}^{(1)}$ требуется $(2N_k + 4)$ процессоров и $(\log_2 N_k + 2)$ тактов.

Тогда для реализации алгоритма обучения максимальное количество необходимых процессоров соответствует величине

$$P_{max} = 8N_k + N_iN_j + 5N_jN_k + 9,$$

а количество временных тактов, необходимых для одной эпохи обучения равно

$$T_{max} = 2\log_2 N_k + 16.$$

Модели производительности параллельного алгоритма обучения трехслойного персептрона представлены в табл. 1 как для декомпозиции алгоритма на уровне реализации функций нейронов (1), (3), так и на уровне реализации скалярных операций (2), (4).

Таблица 1

Описание модели производительности

| | $S_p$ |
|---|---|
| (1) | $\frac{7N_j + 8N_k + 10N_iN_j + 13N_jN_k + 3N_iN_jN_k}{2N_i + 2N_j + 6N_k + 31}$ |
| (2) | $\frac{7N_j + 8N_k + 10N_iN_j + 13N_jN_k + 3N_iN_jN_k}{2\log_2 N_k + 16}$ |
| | $\varepsilon$ |
| (3) | $\frac{7N_j + 8N_k + 10N_iN_j + 13N_jN_k + 3N_iN_jN_k}{L(2N_i + 2N_j + 6N_k + 31)}$ |
| (4) | $\frac{7N_j + 8N_k + 10N_iN_j + 13N_jN_k + 3N_iN_jN_k}{(8N_k + N_iN_j + 5N_jN_k + 9)(2\log_2 N_k + 16)}$ |

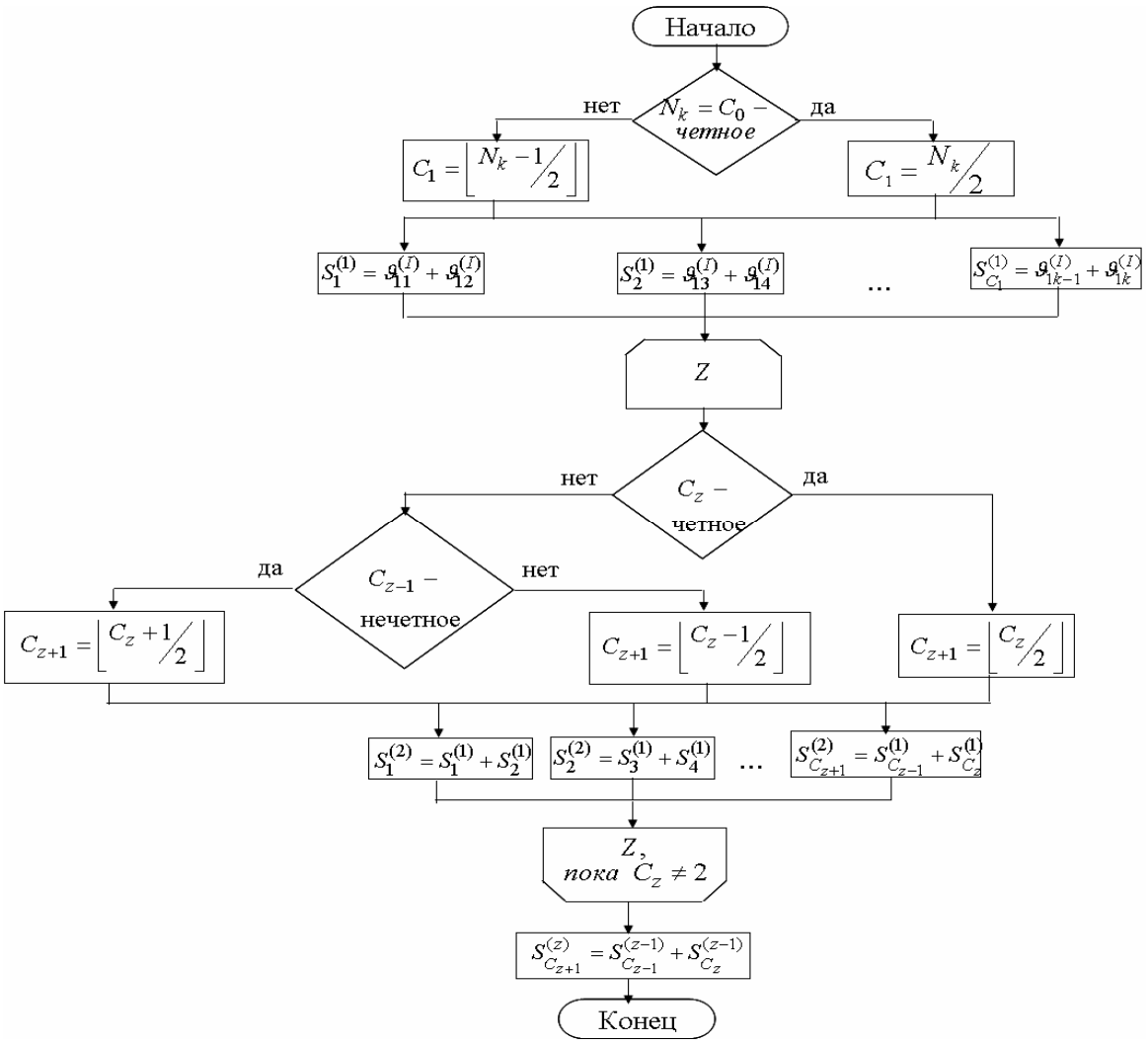


Рис. 2. Параллельная схема вычисления суммы $\vartheta_{jk}^{(II)}$

## Выводы

С целью ускорения алгоритма обучения МП в работе предложен естественный способ декомпозиции нейроалгоритма на уровне скалярных операций.

*Научная новизна* работы заключается в том, что предложены две модели производительности нейроалгоритма для декомпозиции на уровнях реализации функций нейронов и скалярных операций соответственно без учета логической топологии коммуникационных связей вычислительной системы.

Предлагаемые модели производительности позволяют вычислить ускорение и эффективность алгоритма обучения трехслойного персептрона методом обратного распространения ошибки в зависимости от уровня реализации алгоритма, а также определить количество необходимых вычислительных узлов.

## Список литературы

*1. Бухановский А.В., Иванов С.В. Проектирование прикладного математического обеспечения параллельной обработки данных // Учебные материалы Зимней школы- практикума «Технологии параллельного программирования 2006». – СПб. – Нижний Новгород, 2006. – С. 123-129.*

*2. Аксак Н.Г., Тыхун А.Ю. Вычислительная модель нейроалгоритма многослойного персептрона // Высокопроизводительные параллельные вычисления на кластерных системах. Материалы седьмой международной конференции-семинара. – Нижний Новгород, 26–30 ноября 2007. – С. 11-18.*

*Поступила в редколлегию 12.12.2007*

**Рецензент:** д-р техн. наук, проф. Г.Г. Четвериков, Харьковский национальный университет радиоэлектроники, Харьков.