В.А. ГОЛОВКО

НЕЙРОИНТЕЛЛЕКТ: ТЕОРИЯ И ПРИМЕНЕНИЕ

КНИГА 2

Самоорганизация, отказоустойчивость и применение нейронных сетей

Брест 1999

УДК 681.324

Печатается по решению научного совета "Информационные технологии и системы" Института технической кибернетики Национальной Академии наук Республики Беларусь.

Рецензенты:

В.В. Голенков, доктор технических наук, профессор

Р.Х Садыхов, доктор технических наук, профессор

В.Н. Ярмолик, доктор технических наук, профессор

Головко В.А. Нейроинтеллект: теория и применение. Книга 2: Самоорганизация, отказоустойчивость и применение нейронных сетей. Брест Изд. БПИ, 1999 - 228 с.

В книге рассмотрены вопросы проектирования и применения систем искусственного интеллекта, в основе которых лежат нейронные сети. Представлены архитектура и алгоритмы обучения нейронных сетей Кохонена, адаптивного резонанса и гибридных нейронных сетей. Отражены вопросы самоорганизации, отказоустойчивости и реализации нейронных сетей на систолических процессорах. Большое внимание уделено применению нейронных сетей для автономного управления мобильными роботами.

Предназначена для научных работников, специалистов в области вычислительной техники и искусственного интеллекта, а также аспирантов и студентов соответствующих специальностей.

ISBN 985-6584-03-5

ISBN 985-6584-04-3

ПРЕДИСЛОВИЕ

Предлагаемая вниманию читателя книга представляет собой вторую из двух книг, объединенных общим названием "Нейроинтеллект: теория и применение".

В первой книге "Организация и обучение нейронных сетей с прямыми и обратными связями" были рассмотрены следующие нейронные сети: персептронные, рекуррентные, рециркуляционные, Хопфилда, Хэмминга, машина Больцмана и двунаправленная ассоциативная память. В настоящей книге описываются нейронные сети Коххонена, адаптивного резонанса и гибридные нейронные сети. Большое внимание уделено применению нейронных сетей для автономного управления транспортными средствами, реализации нейронных сетей на систолических процессорах, а также самоорганизации и отказоустойчивости нейронных сетей.

В главе 1 рассмотрены самоорганизующиеся сети Кохонена, которые характеризуются обучением без учителя. Они осуществляют топологическое упорядочивание входного пространства образов и основываются на конкурентном обучении. Приводится архитектура и различные алгоритмы обучения таких нейронных сетей. Рассмотрен алгоритм решения задачи коммивояжера с использованием сети Кохонена.

Глава 2 посвящена нейронным сетям адаптивного резонанса. Резонанс в таких сетях происходит при идентификации какого-либо события или образа. Нейронные сети адаптивного резонанса характеризуются обучением без учителя и самоорганизацией в процессе работы. В главе приводятся основные принципы построения и функционирования таких нейронных сетей.

В главе 3 рассматриваются гибридные нейронные сети, которые

представляют собой объединение различного рода нейронных сетей и концепций их обучения. Приведен анализ известных гибридных сетей, таких как нейронные сети встречного распространения и с радиально-базисной функцией активации. Предложены нейронные сети для иерархической классификации образов и решения задач оптимизации.

В главе 4 отражены различные аспекты применения нейронных сетей для автономного управления транспортными средствами. Основное внимание здесь уделено описанию нейронной системы для автономного управления мобильным роботом. Такая система разрабатывается в рамках сотрудничества с лабораторией робототехники (Германия). Рассматриваются основные принципы функционирования нейронной системы и приведены результаты экспериментов.

Глава 5 посвящена реализации нейронных сетей на микроэлектронной технологии. Для этого предлагается отображать нейронные сети на систолические процессоры. Рассмотрены различные варианты реализации нейронных сетей на систолических массивах.

В главе 6 рассматриваются вопросы структурной самоорганизации и отказоустойчивости нейронных сетей, реализованных на систолических процессорах. Предложены различные варианты построения схем, которые характеризуются самоорганизацией и отказоустойчивостью. Приводятся количественные оценки показателей отказоустойчивости при проектировании схемы на уровне кристалла. Рассмотрены примеры проектирования отказоустойчивых нейронных сетей.

Автор стремился изложить материал в доступной для широкого круга специалистов форме и восполнить пробелы в отечественной литературе по данной тематике. Поэтому в монографии подробно рассмотрены различные нейронные сети и приводится много примеров.

Данная работа выполнена в соответствии с INTAS проектами 97-0606

"Development of an intelligent sensing instrumentation structure" и 97-2028 "Intelegent neural system for antonomous control of a mobile robot". Автор благодарит INTAS за поддержку исследований в области нейроинтеллекта.

В заключение автор считает своим долгом выразить глубокую признательность доктору технических наук, профессору В.А. Мищенко за постоянное внимание и поддержку в работе, а так же рецензентам: заведующему кафедрой интеллектуальных информационных технологий БГУИР профессору В.В. Голенкову; заведующему кафедрой ЭВМ БГУИР профессору Р.Х. Садыхову и заведующему кафедрой программного обеспечения и информационных технологий БГУИР профессору В.Н. Ярмолику. Автор выражает искреннюю благодарность профессорам К. Шиллингу и Н. Роту (Германия) за плодотворные дискуссии и предоставление экспериментальной базы для проведения исследований. Автор выражает также свою признательность ректору Брестского политехнического института В.Г. Федорову и заведующему кафедрой ЭВМ Брестского политехнического института Н.В. Кудинову за внимание и поддержку, а также своим коллегам по группе нейронных сетей Брестского политехнического института за помощь в проведении экспериментальных исследований.

ГЛАВА 1. САМОРГАНИЗУЮЩИЕСЯ НЕЙРОННЫЕ СЕТИ КОХОНЕНА

Самоорганизующиеся нейронные сети (self-organising networks) характеризуются обучением без учителя, в результате которого происходит адаптация сети к решаемой задаче. Как уже отмечалось в первой главе (книга 1), к таким сетям относятся нейронные сети Кохонена, адаптивного резонанса и рециркуляционные сети. В каждой из этих сетей самоорганизация происходит в результате различных механизмов обучения. Наиболее известными среди самоорганизующихся нейронных сетей являются сети, которые разработал в 80-х годах финский ученый Кохонен (Kohonen)[1,2-4]. В настоящее время такие сети называют его именем. Нейронные сети Кохонена осуществляют топологическое упорядочивание входного пространства паттернов. Они нашли широкое применение в задачах распознавания образов, оптимизации и управления. В главе рассмотрены архитектура, обучение и функционирование таких нейронных сетей. Приводится алгоритм решения задачи коммивояжера с использованием сети Кохонена.

1.1. Общая характеристика сетей Кохонена

Данные сети позволяют в результате обучения осуществлять топологически непрерывное отображение F входного n-мерного пространства в выходное m-мерное пространство; т.е. $F : R^n \rightarrow R^m$. При этом обучение здесь происходит без учителя на основе образов поступающих на сеть. В качестве метода обучения используется ***конкурентное обучение***. Структура самоорганизующейся нейронной сети представляет собой сеть с прямым распространением сигналов. По мере поступления входных образов на такую

сеть посредством обучения происходит разбиение n-мерного входного пространства на различные области решений, каждой из которых соответствует отдельный нейрон. Границы отдельной области перпендикулярны линиям, проведенным между центроидами соседних областей решений. Такое разделение пространства называется диаграммой Вороного (Voronoi) или картами Кохонена. Для двухмерного случая (n=2, m>n) область решений представляет собой правильные шестиугольники (рис. 1.1), в результате чего получается наименьшая ошибка. Для n>2 наилучшая форма областей решений является неизвестной [5].

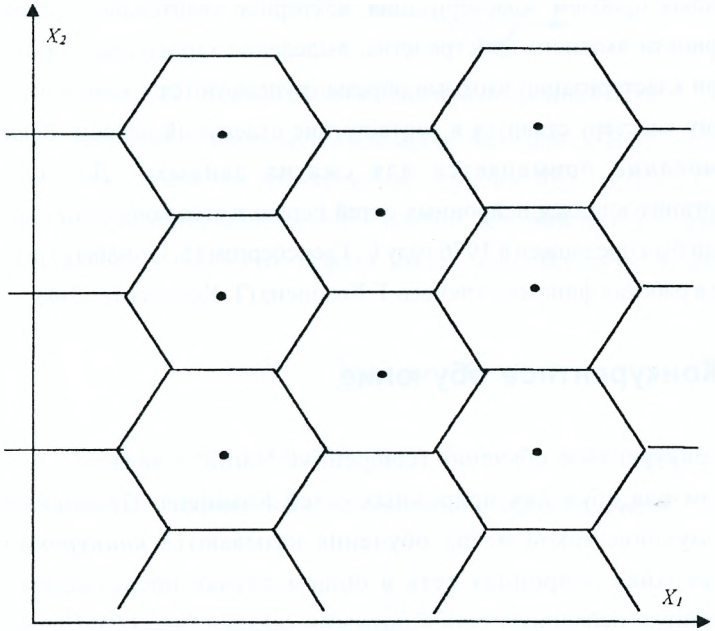


Рис. 1.1. Разбиение входного пространства образов

Таким образом самоорганизация таких сетей происходит в результате топологического упорядочивания входной информации по различным зонам, количество которых является m. Такие зоны или области решений называются кластерами. По аналогии с физикой (книга 1, глава 1) происходит переход от хаоса к порядку.

Топологическое упорядочивание информации напоминает процессы происходящие в головном мозге при его развитии (книга 1, глава 1), когда осуществляется формирование топологически упорядоченных нейронных структур.

Самоорганизующиеся нейронные сети применяются для решения различных проблем: кластеризация, векторное квантование, сокращение размерности входного пространства, выделение характерных признаков и т.д. При кластеризации входные образы группируются в кластеры, причем каждому кластеру ставится в соответствие отдельный нейрон. ***Векторное квантование*** применяется для сжатия данных. Для обучения самоорганизующихся нейронных сетей используется конкурентный метод, который был предложен в 1976 году С. Гроссбергом (S. Grossberg) [6] и затем развит в работах финского ученого Т. Кохонена (T. Kohonen) [1,3-4].

1.2. Конкурентное обучение

Конкурентное обучение (competitive learning) является основным методом для обучения нейронных сетей Кохонена. Нейронные сети, использующие такой метод обучения называются ***конкурентными***. Конкурентная нейронная сеть в общем случае представляет собой двухслойную нейронную сеть с прямыми связями (рис.1.2). Первый слой выполняет чисто распределительные функции, причем каждый нейрон его имеет соединения со всеми нейронными элементами выходного слоя. Во

втором слое происходит конкуренция между нейронными элементами, в результате которой определяется нейрон-победитель.

Для ***нейрона-победителя*** синаптические связи усиливаются, а для

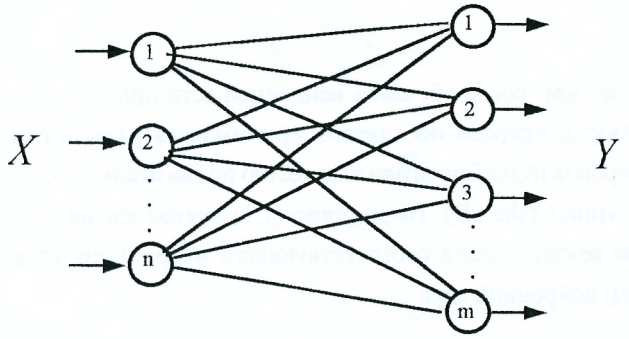


Рис. 1.2. Топология конкурентной нейронной сети

остальных нейронов не изменяются или могут уменьшатся. В результате этого процесса осуществляется конкурентное обучение (competitive learning). Победителем в конкуренции является нейрон, который в результате подачи на вход сети определенного образа имеет максимальную взвешенную активность

$$S_j = \sum_i w_{ij} x_i = W_j^T X \qquad (1.1)$$

где $X = \{x_1, x_2, \ldots, x_n\}$ - входной образ, $W_j = \{w_{1j}, w_{2j}, \ldots, w_{nj}\}$ - вектор столбец весовых коэффициентов j-го выходного нейрона. Пусть

$$S_k = \max_j S_j .$$

Тогда активность выходных нейронов принимается следующей

$$y_j = F(S_j) = \begin{cases} 1, & если \quad k = j \\ 0, & если \quad k \neq j \end{cases} \qquad (1.2)$$

где $j = \overline{1, m}$.

Таким образом, после обучения нейронной сети при подаче входного образа активность нейрона-победителя принимается равной единице, а остальных нейронов нулю. Это правило известно под названием ***«победитель берет все»*** (winner take all). Выражение (1.1) эквивалентно скалярному произведению вектора весов соответствующего нейронного элемента на входной вектор нейронной сети

$$S_j = |W_j| \cdot |X| \cdot \cos\alpha, \qquad (1.3)$$

где $\alpha = \hat{W_j X}$, $|W_j|$ и $|X|$ - модули векторов W_j и X.

Обозначим $P = |X| \cdot \cos\alpha$, где P - проекция вектора X на вектор W. Тогда

$$S_j = |W_j| \cdot P \qquad (1.4)$$

Если векторы $|W_j|$ и $|X|$ ненормированы, то происходит неадекватное определение нейрона победителя (рис. 1.3). Как следует из рисунка нейрон, вектор весов которого W_2 больше отличается от входного образа X чем нейрон имеющий весовой вектор W_1, становится победителем.

Поэтому, при определении нейрона-победителя по взвешенной активности (1.3) необходимо нормализовать весовые и входные векторы для каждого нейрона. Нормализация осуществляется следующим образом:

$$|X| = \sqrt{\sum_i x_i^2} = 1, \qquad (1.5)$$

$$|W_j| = \sqrt{\sum_i w_{ij}^2} = 1. \qquad (1.6)$$

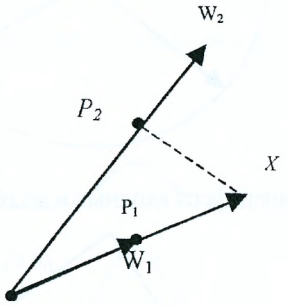


Рис. 1.3. Геометрическая интерпретация определения нейрона победителя при ненормированных весовом векторе и входном образе

Тогда взвешенную активность j-го нейрона можно представить как

$$S_j = |W_j| \cdot |X| \cdot \cos\alpha = \cos\alpha. \qquad (1.7)$$

Из этого выражения следует, что максимальную активность будет иметь тот нейрон, весовой вектор которого коллинеарен входному вектору. Концы векторов при этом находятся на поверхности n-мерной сферы (гиперсферы), радиус которой равен 1(рис. 1.4).

В выражении (1.7) взвешенная сумма эквивалентна коэффициенту взаимной корреляции между входным и весовым вектором. Он будет равен 1, когда угол между векторами равен нулю. Отсюда следует, что правило настройки весовых коэффициентов нейрона-победителя должно соответствовать вращению вектора W_k в сторону вектора X (рис. 1.5).

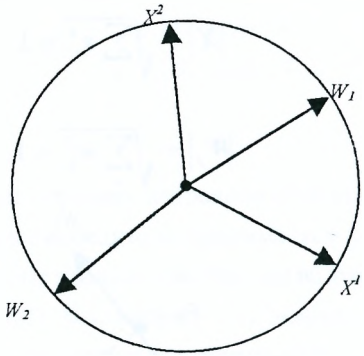


Рис. 1.4. Гиперсфера входных и весовых векторов

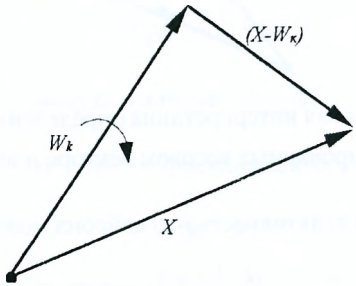


Рис. 1.5. Модификация весового вектора нейрона победителя

В результате можно записать следующее правило обучения для вектора весов нейрона-победителя:

$$W_k(t+1) = W_k(t) + \gamma(X - W_k(t)), \qquad (1.8)$$

где $0 < \gamma < 1$ характеризует скорость обучения.

В качестве нейрона победителя выбирается такой нейрон, весовой вектор которого наиболее близок к входному вектору. В обычной форме

правило обучения для к-го нейрона-победителя можно представить, как

$$w_{ik}(t+1) = w_{ik}(t) + \gamma(x_i - w_{ik}(t)),$$

где $i = \overline{1,n}$.

При применении данного правила к k-му нейрону усиливается его выходная активность (рис. 1.6).

Так как весовой вектор должен быть нормированным, то правило (1.8)

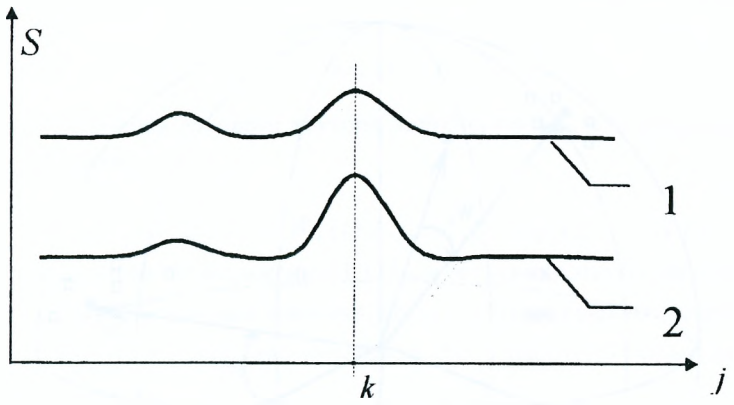


Рис. 1.6. Профили активности нейронов. 1 - профиль активности нейронов до обучения; 2 - профиль активности нейронов после обучения

изменения весового вектора модифицируется следующим образом:

$$W_k(t+1) = \frac{W_k(t) + \gamma(X(t) - W_k(t))}{\left|W_k(t) + \gamma(X(t) - W_k(t))\right|}. \quad (1.9)$$

Соответственно в скалярной форме весовые коэффициенты к-го нейронного элемента определяются, как

$$w_{ik}(t+1) = \frac{w_{ik}(t) + \gamma_i (x_i(t) - w_{ik}(t))}{\left| W_k(t) + \gamma (X(t) - W_k(t)) \right|}. \qquad (1.10)$$

где $i = \overline{1, n}$.

При применении этого правила для обучения нейронной сети весовые векторы нейронов будут вращаться в направлении кластеров входных образов, как показано на рис. 1.7.

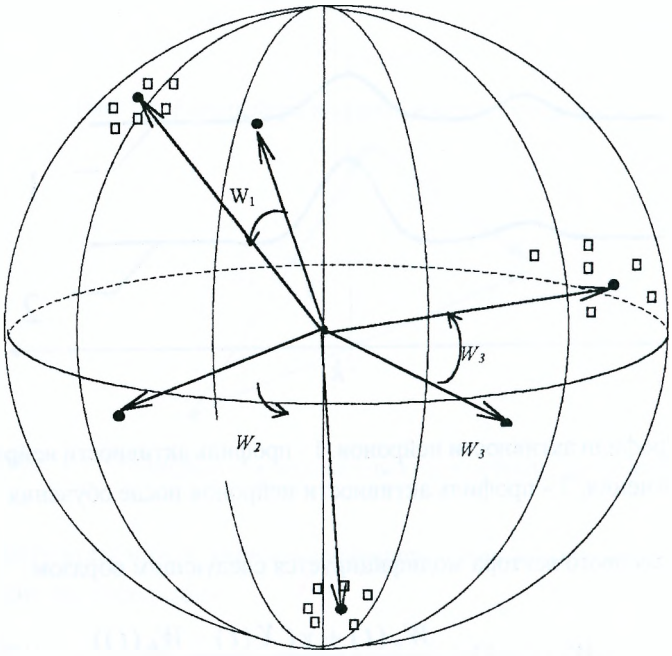


Рис.1.7. Изменение весовых векторов в процессе обучения: □ - вектор паттернов; • - вектор весов.

В случае использования ненормализованных векторов для определения нейрона победителя нужно оперировать вместо взвешенной активности (1.1) евклидовым расстоянием:

$$D_j = \left|X - W_j\right| = \sqrt{(x_1 - w_{1j})^2 + (x_2 - w_{2j})^2 + ... + (x_n - w_{nj})^2} \,. \quad (1.11)$$

При помощи (1.11) определяется нейрон победитель с номером k, который соответствует минимальному евклидовому расстоянию между входным и весовым вектором:

$$D_k = \min_j \left|X - W_j\right| . \quad (1.12)$$

Тогда настройка весового вектора нейрона-победителя происходит следующим образом:

$$W_k(t+1) = W_k(t) + \gamma(X(t) - W_k(t)) \quad (1.13)$$

При использовании выражения (1.13) для обучения одного нейронного элемента на интервале входных значений [a,b] ($x \in [a,b]$) вес нейрона с течением времени стремиться к середине интервала [5]. В работе [7] приводится следующая теорема.

Теорема 1.1: При использовании выражения (1.13) для обучения нейронных элементов происходит минимизация среднеквадратичной ошибки нейрона победителя.

Доказательство: Среднеквадратичная ошибка для κ-го нейрона победителя

$$E = \frac{1}{2}\sum_i (w_{ik} - x_i)^2 . \quad (1.14)$$

Метод градиентного спуска в пространстве весовых коэффициентов минимизирует выражение (1.14). Тогда

$$w_{ik}(t+1) = w_{ik}(t) - \gamma \frac{\partial E}{\partial w_{ik}(t)}.$$

Но

$$\frac{\partial E}{\partial w_{ik}(t)} = (w_{ik} - x_i).$$

Отсюда получаем

$$w_{ik}(t+1) = w_{ik}(t) + \gamma \cdot (x_i - w_{ik}),$$

что и требовалось доказать.

Недостатком описанного выше метода обучения является то, что при случайной инициализации весовых векторов, может получиться так, что некоторые нейроны никогда не будут победителями. Для нейтрализации этого недостатка можно расширить правило обучения следующим образом [7]:

$$\begin{aligned} W_j(t+1) &= W_j(t) + \gamma \cdot (X - W_j(t)), \forall j = k \\ W_j(t+1) &= W_j(t) + \gamma' \cdot (X - W_j(t)), \forall j \neq k \end{aligned}, \qquad (1.15)$$

где $\gamma' << \gamma$. Данное правило позволяет отобразить весовые векторы побежденных нейронов в такую область, где увеличиваются их шансы в конкуренции. Другим вариантом является частотно чувствительное конкурентное обучение (sensitive competitive learning)[7]. Здесь для каждого нейрона ведется статистика его побед. Пусть f_j - частота нахождения j-го нейрона в состоянии победителя. Тогда нейрон победитель определяется следующим образом

$$D_k = \min_j \left| X - W_j \right| f_j. \qquad (1.16)$$

Чем чаще нейрон становится победителем, тем меньше шансов он имеет

в конкуренции.

Итак, при конкурентном обучении все множество входных образов разбивается на кластеры, каждому из которых соответствует свой нейрон. При поступлении на вход нейронной сети неизвестного образа, она будет его относить к такому кластеру, на который он больше всего похож. В этом заключается обобщающая способность такого типа нейронных сетей.

В 1978г. С. Гроссберг доказал сходимость конкурентных методов обучения [8]. Рассмотрим нейронные сети, использующие конкурентные методы обучения.

1.2. Векторный квантователь

Нейронная сеть для векторного квантования была предложена в 1982 г. Кохоненом [1,4]. Векторное квантование используется для сжатия данных и основано на идее сопоставления входного вектора с эталоном [4]. Пусть имеется предварительно сформированное множество эталонных данных, каждое из которых называется кодовым вектором. Совокупность кодовых векторов называется кодовой книгой. При поступлении входного вектора происходит его сравнение с вектором из кодовой книги. В процессе этого выбирается такой кодовый вектор, который наилучшим образом аппроксимирует входной вектор и его номер используется в качестве кода (рис.1.8). В качестве меры близости может использоваться евклидово расстояние D.

Нейронную сеть для векторного квантования принято называть обучающимся векторным квантователем (learning vector quantization). Она представляет собой двухслойную сеть с прямым распространением сигналов, как было показано на рис.1.2. В процессе поступления эталонных векторов на сеть она обучается так, что образуются кластеры различных эталонов, каждому

из которых соответствует свой нейрон. При поступлении на вход такой нейронной сети неизвестного образа, он идентифицируется в соответствии с мерой близости к эталонным векторам и кодируется на выходе сети номером нейрона. Существует большое количество вариантов обучения векторного квантователя [4]. Рассмотрим некоторые из них.

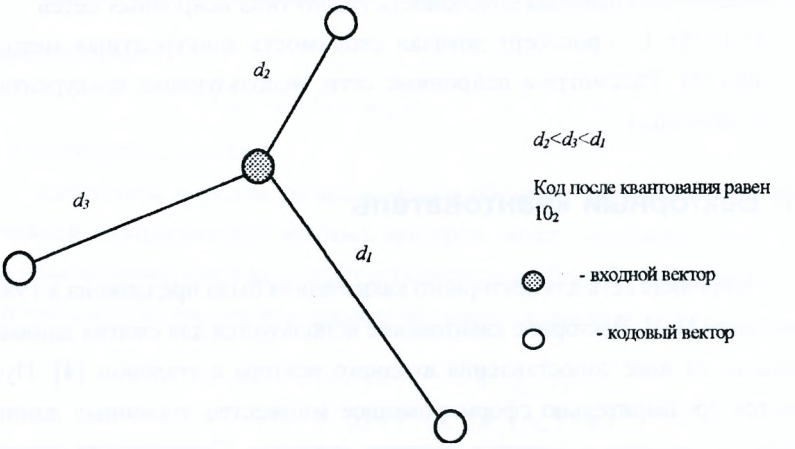


Рис. 1.8. Пример векторного квантования.

1.3.1. Конкурентное обучение с одним победителем

Здесь в отдельный квант времени только один нейрон может быть победителем. Процедура обучения векторного квантователя состоит из следующих шагов:

1. Случайно инициализируются весовые коэффициенты нейронной сети в диапазоне [0,1].
2. Задается начальное значение момента времени $t = 1$.

3. Для всех входных образцов $x^l, l = \overline{1, L}$ последовательно вычисляется:

а) норма вектора

$$D_j^l = \left| X^l - W_j \right|,$$

где $j = \overline{1, m}$;

б) определятся нейрон победитель, обеспечивающий минимальное расстояние

$$D_k^l = \min_j D_j^L;$$

в) производится модификация весовых коэффициентов нейронных элементов

$$w_{ij}(t+1) = w_{ij}(t) + \gamma(t) \cdot (x_i - w_{ij}(t)), \text{ если } j = k$$
$$w_{ij}(t+1) = w_{ij}(t), \text{ если } j \neq k$$

где $i = \overline{1, n}, j = \overline{1, m}$.

Величина $\gamma(t)$ характеризует скорость обучения в момент времени t. Она является постоянной величиной или уменьшается с течением времени по следующему закону:

$$\gamma(t) = \frac{1}{t}.$$

4. Изменяется значение времени $t = t + 1$ и процесс повторяется начиная с п.3.

Обучение производится до получения желаемой степени согласования между входными и весовыми векторами, или до тех пор, пока не перестанут изменяться весовые коэффициенты. В [9] рекомендуется выбирать общее

количество итераций равным

$$t = (50 \div 200)L,$$

где L - общее количество кодовых векторов.

1.3.2 Конкурентное обучение со многими победителями

При конкурентном обучении со многими победителями (Multiple Winner) вокруг нейрона- победителя формируются соседние нейроны, которые также входят в класс победителей V. В этом случае весовые коэффициенты изменяются для всех нейронов, принадлежащих области V. Тогда

$$\Delta w_{ij} = \begin{cases} \gamma(t) \cdot (x_i - w_{ij}), \text{ если } j \in V \\ 0, j \notin V \end{cases} \qquad (1.17)$$

1.3.3. Контролируемое конкурентное обучение

Если заранее известно соответствие эталонных векторов нейронным элементам, то используется контролирусмое конкурентное обучение (Supervised Competitive learning). Для такого обучения весовые коэффициенты нейрона победителя усиливаются при корректной классификации и ослабляется в противном случае. Тогда

$$\Delta W_k = \gamma (X - W_k), \qquad (1.18)$$

если X и W_k принадлежат к одному классу и

$$\Delta W_k = -\gamma (X - W_k), \qquad (1.19)$$

если X и W_k принадлежат к различным классам. Весовые коэффициенты остальных нейронов при этом не изменяются.

После обучения нейронная сеть может осуществлять функции векторного квантования. При этом на выходе в каждый квант времени будет активным только один нейрон (его выход равен 1), а остальные нейроны будут иметь нулевые выходные значения.

1.4. Самоорганизующиеся карты Кохонена

Самоорганизующиеся карты Кохонена (self organization maps) являются дальнейшим расширением нейронных сетей с конкурентным обучением [3,4,9]. Они были разработаны Кохоненом в 1982 году. Топология нейронной сети Кохонена состоит из двух слоев. Первый слой выполняет распределительные функции, а нейроны второго слоя расположены на плоскости образуя матрицу (рис. 1.9).

В отличии от обучающегося векторного квантователя здесь каждый нейрон второго слоя связан соединениями со своими ближайшими соседями. Сила связей между нейронами второго слоя обычно зависит от расстояния между ними и может определяться различными функциями. Так, например, одна из распространенных функций имеет вид «мексиканской шляпы» (рис.1.10), где близко расположенные нейроны соединены усиливающими связями. С увеличением расстояния усиление сменяется торможением, а затем опять появляются слабо возбуждающие связи. Таким образом каждый нейрон второго слоя имеет здесь два вида связей: w_{ij} - вектор связей, поступающий от входного слоя и v_j - вектор связей поступающий на j-ый нейрон от соседних нейронов второго слоя. Тогда выходная активность j-го нейрона [5]

$$y_j(t) = F\left(\sum_i w_{ij} x_i + \sum_{i \neq j} v_{ij} y_i(t)\right), \qquad (1.20)$$

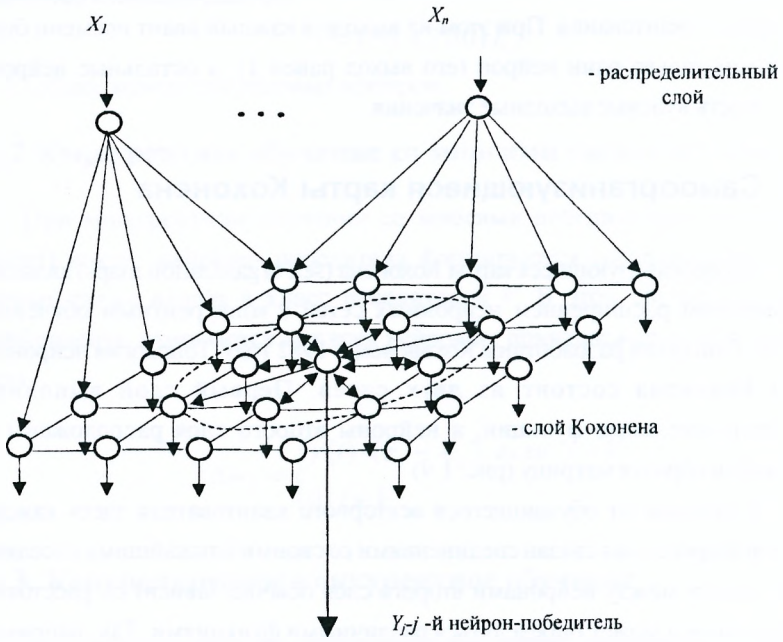


Рис. 1.9. Топология нейронной сети Кохонена

где F - функция нелинейного преобразования. В качестве функции F здесь обычно применяется сигмоидная функция. В результате победителем здесь является не один нейрон, а группа нейронов. Латеральные связи, ослабляющие синаптические соединения, способствую наличию контрастной границы раздела между активными и остальными нейронами. В выражении (1.20) $v_{ij} = const$. В процессе релаксации сети в соответствии с выражением (1.20) происходит усиление выходной активности у нейрона-победителя и его ближайшего окружения (рис. 1.6). Выходные значения остальных нейронных элементов ослабляются. Для обучения сети Кохонена с латеральными связями применяется следующее правило [5]:

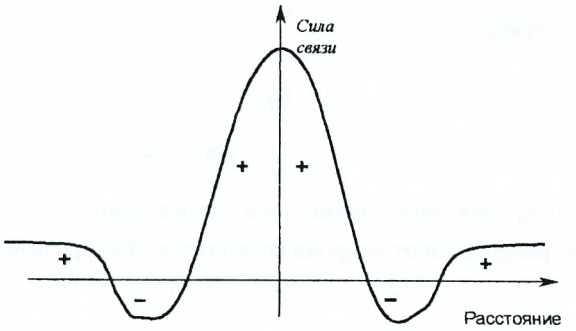


Рис 1.10. Изменение силы связи в зависимости от расстояния между нейронами

$$w_{ij}(t+1) = w_{ij}(t) + \gamma(x_i y_j - \beta(y_j) w_{ij}(t)), \quad (1.21)$$

где при помощи терма "$-\beta(y_j)w_{ij}(t)$" происходит предотвращение неограниченного возрастания весов.

Как уже отмечалось, в такой нейронной сети победителем является не один, а группа нейронных элементов. Это используется для популяционного кодирования (Population Coding) информации [5].

В своем простейшем виде сеть Кохонена функционирует по принципу «победитель берет все». При этом она должна выполнять топологически упорядоченное отображение входных векторов на матрицу нейронов второго слоя. Соседние наиболее похожие входные образы должны отображаться на соседние нейроны матрицы. Это достигается путем введения области притяжения G для нейрона победителя, в радиусе действия которой нейроны активно изменяют свои весовые векторы в сторону входного образа. Область притяжения можно описать функцией притяжения, которую обозначим $h(t,k,p)$. Здесь t -время, k - номер нейрона победителя, p - номер искомого

нейрона. В дискретном варианте функция притяжения определяется следующим образом:

$$h(t,k,p) = \begin{cases} 1, & если \quad p \in G \\ 0, если & p \notin G \end{cases} \qquad (1.22)$$

В область притяжения нейрона k входят все нейроны находящиеся на определенном расстоянии от нейрона победителя. В непрерывном варианте

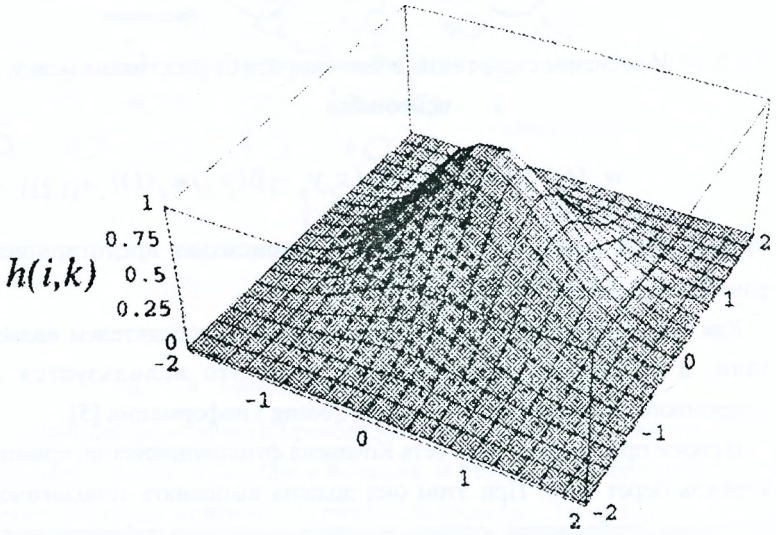


Рис. 1.11. Двухмерная функция Гаусса

часто используется функция Гаусса (рис. 1.11).

Она определяется при помощи следующего выражения:

$$h(p,k,t) = e^{\frac{-(u_k - u_p)^2}{2\sigma^2(t)}}, \qquad (1.23)$$

где $(u_k - u_p)$ - расстояние между нейронами, σ(t) - среднеквадратичное отклонение (радиус области притяжения). Положение каждого нейрона в матрице характеризуется его координатами:

$$u_k = (i_k, j_k), u_p = (i_p, j_p). \qquad (1.24)$$

Тогда

$$(u_k - u_p)^2 = (i_k - i_p)^2 + (j_k - j_p)^2. \qquad (1.25)$$

В процессе обучения нейронной сети Кохонена изменяются весовые коэффициенты не только нейрона победителя, но и всех нейронов внутри области притяжения. Так для *p*-ого нейрона весовой вектор изменяется по следующему закону:

$$W_p(t+1) = W_p(t) + \gamma(t)h(t,k,p)(X(t) - W_p(t)). \qquad (1.26)$$

С увеличением времени обучения радиус области притяжения уменьшается. В результате нейронные элементы сжимаются около нейрона победителя, пока он не останется один. Это схематично изображено на рис. 1.12, где $G(t)$ - область притяжения в момент времени *t*. Введем декартову систему координат, так, что каждый нейрон имеет координаты (i, j), где $i = \overline{1,m}; j = \overline{1,m}$ (рис. 1.12).

Поставим в соответствие нейрону с координатами (i,j) весовой вектор W_{ij}, который определяется следующим образом:

$$W_{ij} = (w_{1ij}, w_{2ij}, ..., w_{nij}), \qquad (1.27)$$

где n – количество нейронных элементов входного слоя.

Тогда процедуру обучения сети Кохонена для непрерывной функции притяжения можно представить в виде следующей последовательности действий:

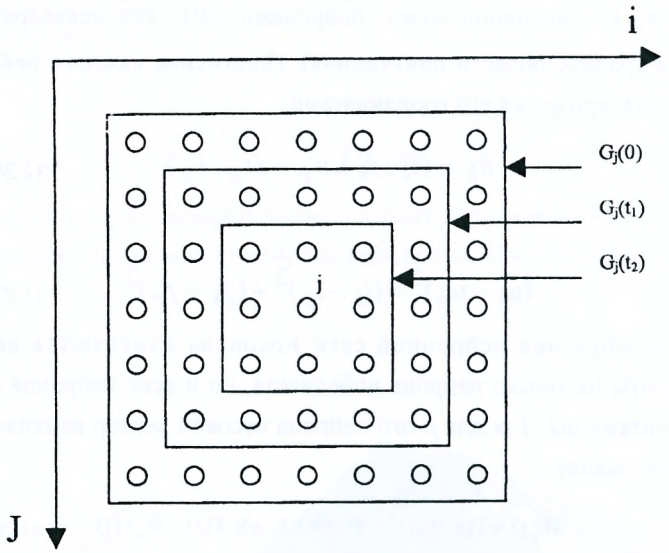


Рис. 1.12. Изменение области притяжения с течением времени

1. Случайно инициализируются весовые коэффициенты W нейронной сети.

2. Задается начальное значение радиуса притяжения σ и момент времени $t = 1$.

3. Подается входной образ $X^l (l = 1)$ на нейронную сеть и вычисляется норма вектора:

$$D_{ij} = (X^l - W_{ij}),$$

где $i = \overline{1,m}$, $j = \overline{1,m}$, W_{ij} - весовой вектор нейрона с координатами (i, j).

4. Определяется нейрон победитель:

$$D(k_1, k_2) = \min_{i,j} D_{ij},$$

где $k = (k_1, k_2)$ - координаты нейрона победителя

5. Для каждого нейрона производится вычисление функции притяжения:

$$h(p,k,t) = e^{\frac{-(u_k - u_p)^2}{2\sigma^2(t)}},$$

где $p = \overline{1, m^2}$.

6. В соответствии с функцией притяжения осуществляется модификация весовых коэффициентов:

$$W_{ij}(t+1) = W_{ij}(t) + \gamma(t)h(p,k,t)(X^l(t) - W_{ij}(t)),$$

где $\gamma(t) = \frac{1}{t}$.

7. Переходим к пункту 3 и повторяем процедуру для $l = 2,3,..,L$, где L-общее количество входных образов.

8. Увеличиваем на единицу квант времени, уменьшаем радиус области притяжения σ и повторяем процесс начиная с пункта 3.

Обучение производится до получения желаемой степени согласования между весовыми и выходными векторами. Начальное значение области притяжения σ может охватывать всю матрицу нейронов, а затем последовательно уменьшается, как показано на рис. 1.12. Для дискретной функции притяжения процедура обучения является аналогичной. Функционирование такой сети происходит путем определения нейрона победителя и присвоения ему единичного значения, а остальным нейронам нулевого значения.

В процессе обучения происходит упорядочивание весовых

коэффициентов таким образом, что уменьшается разница между весами соседних нейронов. Покажем это. Пусть весовые векторы соседних нейронов изменяется следующим образом:

$$W_1(t+1) = W_1(t) + \gamma(x(t) - W_1(t)) = (1-\gamma)W_1(t) + \gamma X(t)$$
$$W_2(t+1) = W_2(t) + \gamma(x(t) - W_2(t)) = (1-\gamma)W_2(t) + \gamma X(t)$$

Расстояние между ними равно

$$|W_1(t+1) - W_2(t+1)| = |(1-\gamma)(W_1(t) - W_2(t))|.$$

Так как $0 < \gamma < 1$, то

$$|W_1(t+1) - W_2(t+1)| < |W_1(t) - W_2(t)|.$$

Рис. 1.13. Изменение весовых коэффициентов сети Кохонена

Таким образом в процессе обучения разница между весовыми векторами топологически близких нейронов уменьшается. Это показано на рис.1.13 для сети с двумя входными нейронами и 8x8 выходными нейронами. Линии соединяют значения весов нейрона с координатами (i,j) с весами нейрона (i+1, j) и (i, j+1). В начале веса выбраны случайно и беспорядочно распределены на плоскости. Затем веса изменяются, так что их плотность приблизительно соответствует плотности вероятности входных векторов.

1.5. Решение задачи коммивояжера

Пусть количество городов, которое должен объехать коммивояжер, равняется N. При использовании полного перебора, общее количество вариантов, которое необходимо просмотреть для решения данной задачи равняется (N-1)!.

Пусть известны координаты каждого города (x,y). Они являются входными данными для нейронной сети Кохонена. Таким образом, совокупность координат всех городов образует обучающее множество. Архитектура нейронной сети состоит из двух входных нейронов, которым соответствуют координаты x и y и из N нейронов слоя Кохонена. Нейроны слоя Кохонена образуют одномерную цепочку (рис. 1.14).

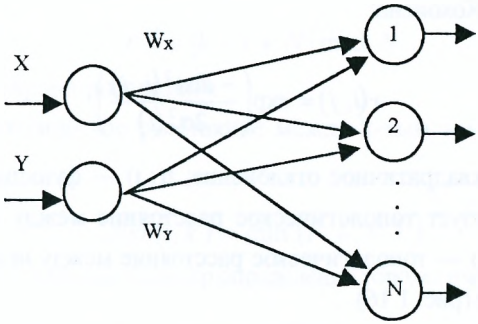


Рис. 1.14. Архитектура сети Кохонена для решения задач коммивояжера

Функционирование такой нейронной сети для решения задачи коммивояжера основывается на том, что при обучении города, расположенные по соседству друг с другом, будут отображаться на нейроны, расположенные по соседству в слое Кохонена. После обучения сети города должны образовывать замкнутую цепочку (рис. 1.15).

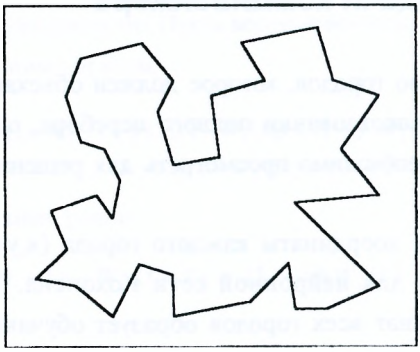


Рис. 1.15. Формирование замкнутого маршрута сетью Кохонена

В соответствии с этим введем следующую функцию притяжения между нейронами в слое Кохонена:

$$r(i,j) = \exp\left(\frac{-dist^2(i,j)}{2\sigma^2(t)}\right),$$

где σ(t) — среднеквадратичное отклонение; r(i,j) — функция притяжения, которая характеризует топологическое расстояние между i-тым и j-тым нейронами; dist(i,j) — топологическое расстояние между нейронами i и j в замкнутой цепочке (рис. 1.16)

Для определения dist(i,j) необходимо вначале определить топологическое расстояние между i-тым и j-тым нейронами, двигаясь по окружности как по часовой, так и против часовой стрелки. Затем необходимо взять наименьшее получающееся число, которое и является искомым топологическим расстоянием между соответствующими нейронами. Топологическое расстояние при движении по окружности в направлении часовой стрелки определяется следующим образом:

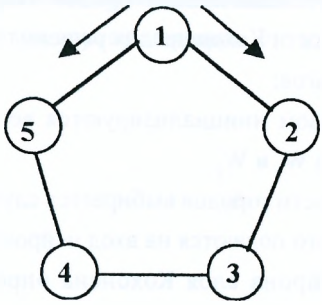


Рис. 1.16. Пример замкнутого маршрута для пяти городов

$$P\downarrow = |k - i - N| \bmod N \tag{1.29}$$

Соотвстственно, при движении против часовой стрелки:

$$P\uparrow = |k - i + N| \bmod N\ , \tag{1.30}$$

где k — номер нейрона-победителя.

Тогда топологическое расстояние между k-тым и i-тым нейронами равняется:

$$dist(i,k) = \min\left(P\downarrow, P\uparrow\right). \tag{1.31}$$

В таблице 6.1 приведен пример определения топологического расстояния

Таблица 1.1

i	$P\downarrow$	$P\uparrow$	$dist\ (i,k)$
1	0	0	0
2	1	4	1
3	2	4	2
4	3	2	2
5	4	1	1

для цепочки нейронов, состоящих из пяти городов, в случае k=1 (рис. 1.16).

Алгоритм обучения сети Кохонена для решения задачи коммивояжера состоит из следующих шагов:

1. Случайным образом инициализируются весовые коэффициенты нейронов в слое Кохонена W_x и W_y.

2. Из всей совокупности городов выбирается случайным образом один город, координаты которого подаются на вход нейронной сети.

3. Для каждого нейрона слоя Кохонена определяется евклидово расстояние между координатами выбранного города и соответствующими весовыми коэффициентами:

$$D_j = (x - w_{xj})^2 + (y - w_{yj})^2,\ j = \overline{1, N},$$

4. Определяется нейрон-победитель, который обеспечивает наименьшее ьшее евклидово расстояние:

$$D_k = \min_j \{D_j\},$$

где k — номер нейрона-победителя.

5. В соответствии с функцией притяжения r(j,k) модифицируются весовые коэффициенты нейронной сети:

$$w_{xj}(t+1) = w_{xj}(t) + r(j,k)\gamma(t)(x - w_{xj}(t)),$$
$$w_{yj}(t+1) = w_{yj}(t) + r(j,k)\gamma(t)(y - w_{yj}(t)),$$

где $J = \overline{1, N}$.

6. Переходим к пункту 2 и повторяем указанную процедуру для всех городов.

7. Уменьшаем значение γ(t), σ(t) и повторяем процесс, начиная с пункта 2.

Данная процедура продолжается до тех пор, пока не перестанут изменяться весовые коэффициенты нейронной сети. После обучения,

каждому нейрону слоя Кохонена будет поставлен в соответствие определенный город, которые образуют замкнутый маршрут. Это и будет являться решением задачи коммивояжера.

ГЛАВА 2. НЕЙРОННЫЕ СЕТИ АДАПТИВНОГО РЕЗОНАНСА

Нейронные сети адаптивного резонанса были предложены С. Гроссбергом (S. Grossberg) в 1976 году [10]. Они основываются на теории ***адаптивного резонанса*** (Adaptive Resonance Theory). В соответствии с ней такие нейронные сети называются ART сетями. Резонанс в них происходит при идентификации какого-либо события или образа. В процессе функционирования ART сетей в них происходит циркуляция информации до тех пор пока не наступит состояния резонанса. Нейронные сети адаптивного резонанса обучаются без учителя и характеризуются самоорганизацией в процессе работы. Они могут использоваться для распознавания образов, обработки речевых сигналов и в задачах управления. В главе описываются основные принципы построения и функционирования нейронных сетей адаптивного резонанса [10,11,7,12,13].

2.1. Основы адаптивного резонанса

Теория адаптивного резонанса базируется на следующих основных принципах [7]:

1. Адаптация входного паттерна к паттернам хранящимся в сети осуществляется при помощи резонанса.

2. Резонанс происходит при идентификации входного образа, когда он максимально совпадает с образом хранящимся в сети. В процессе функционирования сеть резонирует до тех пор, пока не выделит паттерн наименее отличающийся от входного или не зарезервирует новый класс.

3. В процессе адаптации входного паттерна к образам имеющимся в

сети происходит его контрастное усиление. Оно характеризуется тем, что только отличительные особенности входного паттерна отображаются на синаптические веса. Это напоминает процесс биологической эволюции, когда отдельные признаки усиливаются, а другие ослабляются.

4. Наличие ***кратковременной*** (short-term-memory) и ***долговременной*** (long-term memory) памяти. Кратковременная память хранит входной паттерн, который должен быть декодирован, а долговременная память соответствует образам, которые хранятся в нейронной сети.

Отсюда следует, что теория адаптивного резонанса имеет биологические предпосылки. Так долговременная и кратковременная память соответствует типам памяти, которые имеются у человека. Явление резонанса, как отмечалось в главе 1 тома 1, играет большую роль как при самоорганизации индивида, так и биологической системы.

Постановка задачи при распознавании образов методом адаптивного резонанса состоит в следующем. Необходимо найти такие синаптические векторы $W_1, W_2, \ldots W_m$, которые разбивают входное пространство паттернов на различные кластеры. Каждый кластер имеет размер, который характеризуется угловым расстоянием α (рис. 2.1) и соответствующей ему величиной ρ. Величина ρ называется ***параметром бдительности***, который равен:

$$\rho = \cos\alpha$$

Если ρ имеет маленькое значение, то входные векторы будут отображаться на большие кластеры, в противном случае на маленькие (рис. 2.1).

В соответствии с параметром бдительности нейронная сеть должна каждый раз решать, принадлежит ли входной вектор уже имеющемуся кластеру или резервировать для него новый кластер с соответствующим весовым вектором. Это обеспечивает с одной стороны ***пластичность*** сети,

так как каждый раз сеть может реагировать на незнакомый образ и с другой стороны ***стабильность***, так как уже идентифицированные кластеры не размываются посредством новых входных векторов [14].

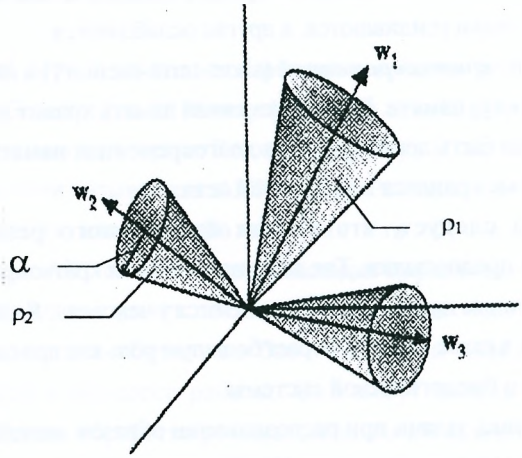


Рис.2.1. Разбиение входного пространства образов на кластеры: ρ - параметр бдительности; $\rho_1 < \rho_2$

Существуют различные модели нейронных сетей, основанных на теории адаптивного резонанса. В общем случае архитектура таких сетей представлена на рис. 2.2 . Она состоит из двух слоев, которые соединены между собой прямыми и обратными синаптическими связями. Весовой вектор W характеризует прямые синаптические связи, а вектор V – обратные синаптические связи (рис. 2.2). Информация хранящаяся в этих связях характеризует долговременную память (LTM). Активизация нейронов каждого из слоев характеризует кратковременную память (STM). При помощи конкурентного слоя происходит отображение входного паттерна в соответствующий кластер.

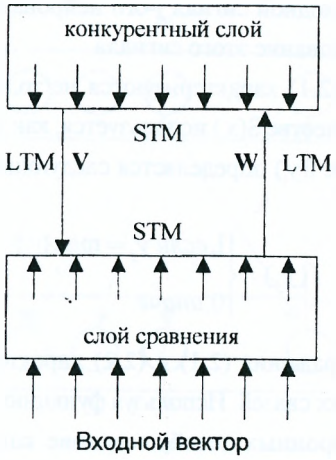


Рис. 2.2. Общая структура ART сети

Сравнивающий слой анализирует степень совпадения входного и выделенного сетью образа. При достаточной степени совпадения наступает резонанс, что соответствует идентификации образа.

Для описания функционирования и обучения таких сетей С. Гроссберг использовал дифференциальные уравнения [8]:

$$\frac{dv_{ji}}{dt} = \alpha_1 f(y_j)[-\beta_1 v_{ji} + S(x_i)], \qquad (2.1)$$

$$\frac{dw_{ji}}{dt} = \alpha_2 f(y_j)[-\beta_2 w_{ij} + S(x_i)], \qquad (2.2)$$

где α_1 и α_2 положительные константы, управляющие скоростью обучения; β_1 и β_2- положительные константы, определяющие торможение связей; v_{ji} и w_{ij} - синаптические связи; x_i - i-ая компонента входного образа X; $S(x_i)$ - нелинейное

преобразование x_j; y_i - выходной сигнал j-ого нейрона конкурентного слоя; $f(y_j)$- нелинейное преобразование этого сигнала.

Выражения (2.2) и (2.1) характеризуются не только усилением, но и торможением связей. В качестве $S(x_i)$ используется, как правило, сигмоидная функция. Преобразование $f(y_j)$ определяется следующим образом:

$$f(y_j) = \begin{cases} 1, если\ y_j = \max\{y_i\} \\ 0, иначе \end{cases} \qquad (2.3)$$

Таким образом, выражения (2.1) и (2.2) характеризуются активным забыванием синаптических связей. Используя функцию Ляпунова, Гроссберг доказал сходимость нейронных сетей, в основе которых лежит теория адаптивного резонанса [12].

2.2. Архитектура нейронной сети адаптивного резонанса

Гроссберг разработал ряд архитектур ART сетей. В данном разделе рассматривается одна из таких архитектур, которая называется ART1 сетью [10]. Она предназначена для обработки бинарных векторов и состоит из двух слоев нейронных элементов F1 и F2 (рис. 2.3). Слой F1 содержит n нейронных элементов, где n - размерность входного образа. Каждый нейронный элемент при этом имеет синаптические связи со всеми элементами слоя F2. Такие синаптические связи являются прямыми и характеризуются весовыми векторами $W=(W_1, W_2, \ldots W_m)$, где m - количество нейронов слоя F2. Весовой вектор $W_i=(w_{1i}, w_{2i}, \ldots w_{ni})$ соответствует весовым коэффициентам i-ого нейрона второго слоя. Каждый нейрон слоя F2 характеризует определенный кластер образов и имеет синаптические связи со всеми нейронами слоя F1.

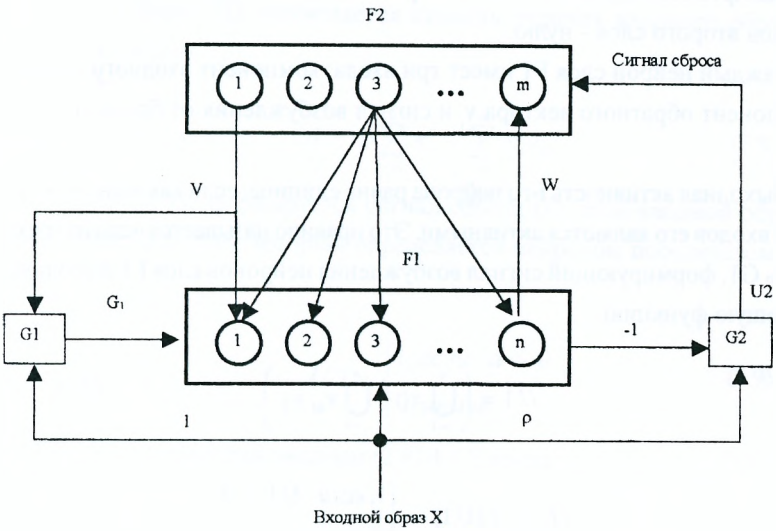


Рис. 2.3. Архитектура ART1 сети

Эти синаптические связи называются обратными и характеризуются весовыми векторами $V=(V_1, V_2 \dots V_m)$. При этом вектор $V_j=(v_{j1}, v_{j2} \dots v_{jn})$ отражает весовые связи j-го нейрона слоя F1. Прямой весовой вектор W_i соответствует i-му кластеру, а обратный весовой вектор V_i характеризует интегрированный образ соответствующий этому кластеру. Активность нейронов в слое F1 и F2 символизирует кратковременную память, а весовые векторы W и V - долговременную память. Нейронные элементы слоя F2 функционируют по принципу «победитель берет все» (winner take all). При этом определяется нейрон-победитель, который имеет максимальное значение скалярного произведения весового на входной вектор

$$y_k = \max_j \left(W_j^T X\right). \qquad (2.4)$$

Выход нейрона победителя становится равным единице, а выходы остальных нейронов второго слоя - нулю.

Каждый нейрон слоя F1 имеет три входа: компонент входного вектора x_i, компонент обратного вектора v_i и сигнал возбуждения от блока G1 (рис. 2.3).

Выходная активность i-го нейрона равна единице, если как минимум два из трех входов его являются активными. Это правило называется «два из трех». Модуль G1, формирующий сигнал возбуждения нейронов слоя F1 выполняет следующую функцию:

$$U1 = \left(\bigcup_{i=1}^{n} x_i - \bigcup_{i=1}^{n} v_{ki} y_k \right), \tag{2.5}$$

$$G_1 = f(U1) = \begin{cases} 1, \textit{если} \quad U1 > 0 \\ 0, \textit{иначе} \end{cases} \tag{2.6}$$

где y_k - выходное значение нейрона победителя второго слоя; $\bigcup$ – операция логическое “ИЛИ”; G_1 – выходное значение сигнала модуля G1.

Каждый нейрон слоя F1 содержит два выхода. Один выход соответствует компоненте входного вектора x_i, которая через прямые связи поступает на слой F2. Второй выход z является результатом сравнения входного x_i и обратного v_{ki} сигналов:

$$z_i = x_i v_{ki}. \tag{2.7}$$

Модуль G2 в соответствии с параметром бдительности r формирует сигнал сброса, который нейтрализует активный нейронный элемент слоя F2. На входы G2 поступают сигналы z_i через синаптические связи равные **-1** и компоненты входного образа $X=\{x_1, x_2 \dots x_n\}$ через связи с весовыми коэффициентами r. В

результате в блоке G2 вычисляется степень отличия входного образа от хранящегося в сети

$$d = \rho \sum_{i=1}^{n} x_i - \sum_{i=1}^{n} x_i v_{ki}. \qquad (2.8)$$

На выходе G2 формируется сигнал сброса U2, если входной образ не соответствует кластеру, который определяется нейроном победителем. Тогда

$$U2 = \begin{cases} 1, \textit{если} \;\; d > 0 \\ 0, \textit{иначе}. \end{cases} \qquad (2.9)$$

В соответствии с выражением (2.8) d > 0, когда

$$\rho > \frac{\sum_{i=1}^{n} x_i v_{ki}}{\sum_{i=1}^{n} x_i} \qquad (2.10)$$

2.3. Функционирование ART1 сети

Пусть в начальный момент времени на слой F1 поступает входной вектор X. В результате, согласно выражению (2.6), модулем G1 формируется сигнал возбуждения G_1 нейронов слоя F1. Сигнал y_k=0, так как в начальный момент времени еще не определен нейрон победитель. В соответствии с правилом «два из трех» осуществляется активация нейронов первого слоя, что соответствует записи входного образа в кратковременную память. Производится распространение входного вектора X через прямые

синаптические связи W к нейронам второго слоя, где определяется нейрон победитель. Выходной сигнал нейрона победителя $y_k=1$ через обратные синаптические связи поступает на нейроны слоя F1.В результате формируется интегрированный паттерн активности $V_k=(v_{k1},v_{k2} \dots v_{kn})$, который соответствует кластеру нейрона победителя. Активность нейрона победителя $y_k=1$ блокирует сигнал возбуждения от модуля G1, выходное значение которого становиться равным нулю ($G_1=0$). Поэтому на входы нейронов слоя F1 поступают два вектора: входной вектор X и обратный вектор V_k. Согласно выражению (2.7) вычисляется активность нейронов первого слоя $Z=(z_1, z_2 \dots z_n)$. Если отношение

$$\frac{|Z|}{|X|}=\frac{\sum_{i=1}^{n} z_i}{\sum_{i=1}^{n} x_i}<\rho,$$

то генерируется сигнал сброса, который выключает нейрон победитель из дальнейшей конкуренции. Это приводит к блокированию обратного вектора V_k, в результате чего в кратковременной памяти F1 восстановится первоначальный входной вектор. Процесс повторяется до тех пор, пока входной паттерн не станет достаточно похожим на один из паттернов, которые хранятся в сети (состояние резонанса). В противном случае для входного паттерна резервируется новый нейрон слоя F2. В любом случае нейронная сеть достигает стабильного состояния, после которого может происходить процесс обучения. Итак, процедуру функционирования такой нейронной сети можно представить в общем случае как следующую последовательность шагов:

1. На входы нейронов слоя F1 подается образ $X=\{x_1, x_2, \dots, x_n\}$

2. Через прямые связи w_{ij} входной образ поступает на нейроны слоя F2.

3. Каждый нейрон слоя F2 состязается с соседними, пока не останется активным только один.

4. Победивший нейронный элемент из слоя F2 посылает единичный сигнал сверху вниз по обратным связям v_{ji} к нейронам первого слоя F1.

5. Входной образ $X=\{x_1, x_2, ..., x_n\}$ сравнивается с образом, который определяется нейроном победителем слоя F2 посредством весовых коэффициентов v_{ji}

6. Если отличие между двумя образами не превышает значение, определенное параметром бдительности, то победивший нейронный элемент неадекватно отражает входной образ и происходит его удаление из дальнейшей конкурентной борьбы. Дальнейшие действия в этой ситуации сводятся к следующим шагам:

а) если еще остались нейронные элементы в слое F2, которые не были победителями, то перейти к шагу 2 алгоритма;

б) если таких нейронных элементов не существует, то необходимо зарезервировать свободный нейрон из слоя F2 и произвести настройку его синаптических весов в соответствии с входным образом;

в) если не существует свободных нейронов из слоя F2, то прекратить выполнение алгоритма и выдать сообщение о невозможности идентификации образа.

7. Если отличие между прямым и обратным образом превышает значение параметра бдительности, то победивший нейронный элемент определяет правильный класс для входного образа. Выходное значение нейронной сети содержится в весовых коэффициентах v_{ki} нейрона победителя, значения которых характеризуют интегрированный образ кластера.

2.4. Алгоритм обучения и функционирования ART1 сети

ART1 сеть характеризуется тем, что процессы обучения и функционирования неразрывно связаны между собой. С. Гроссберг описывал такую адаптивную сеть при помощи дифференциальных уравнений для непрерывного времени [10]. В 1987 г. Р. Липманн (R. Lippmann) предложил алгоритм работы ART1 сети для дискретного случая [11]. Данный алгоритм представляет собой следующую последовательность действий:

1. Инициализация прямых и обратных синаптических связей:

$$w_{ij}(0) = 1/(1+n),$$
$$v_{ij}(0) = 1,$$

где $i = \overline{1,n}$; $j = \overline{1,m}$.

2. Установка параметра бдительности:

$$0 < \rho <= 1.$$

3. Подача на вход неизвестного образа $X=\{x_1, x_2, ..., x_n\}$

4. Вычисление выходной активности нейронных элементов второго слоя:

$$y_i = \sum_{j=1}^{n} w_{ji}(t) x_i, \quad \forall i = \overline{1,m}$$

5. Определение нейрона победителя с номером k:

$$y_k = \max_i \{y_i\}$$

6. Определение теста контроля принадлежности входного образа к выделенному k-му кластеру. Если

$$\frac{\sum_{i=1}^{n} v_{ki}(t)x_i}{\sum_{i=1}^{n} x_i} > \rho,$$

то перейти к шагу 8. В противном случае перейти к шагу 7.

7. Производится удаление нейрона-победителя с номером *к* слоя F2 из конкурентной борьбы. В этом случае установить $y_k=0$ и если существуют нейроны в слое F2, которые не были нейронами-победителями, то перейти к шагу 4 алгоритма. Если таких нейронов не существует, то необходимо зарезервировать свободный нейрон в слое F2 для данного входного образа или выдать сообщение о невозможности идентификации.

8. Для текущего входного образа производится настройка прямых и обратных связей в соответствии со следующими выражениями:

$$v_{ki}(t+1) = v_{ki}(t)x_i,$$

$$w_{ik}(t+1) = \frac{v_{ki}(t)x_i}{0.5 + \sum_{i=0}^{n} v_{ki}x_i},$$

где $i = \overline{1,n}$.

При функционировании такой сети, каждый кластер будет интегрировать признаки, которые являются общими для входных образов, принадлежащих этому кластеру. Так, например, если настроить нейронную сеть на букву «О» и затем подать на сеть букву "Q" (рис.2.4 а, б), то нейронная сеть будет стараться адаптироваться к новому образу и не забыть старый. В результате она запомнит образ изображенный на рис. 2.4 б.

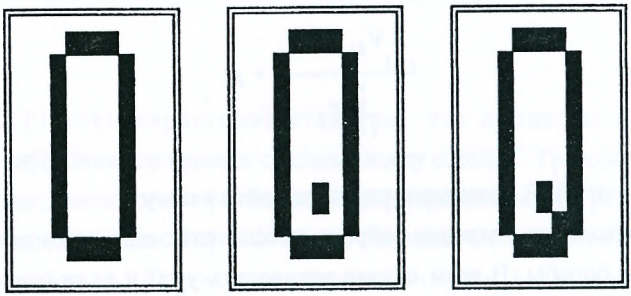


Рис. 2.4. Обучение ART сети похожим паттернам

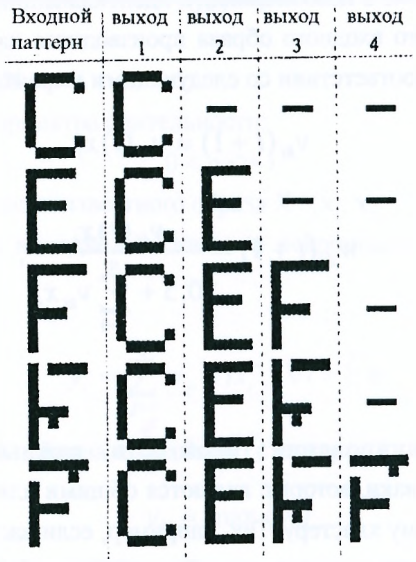


Рис. 2.5. Обучение ART сети пяти буквам

Если еще раз на вход сети подать букву «Q», то старый образ (рис.2.4а) полностью трансформируется в образ буквы «Q» (рис. 2.4в). На рис.2.5 изображен пример последовательного запоминания нейронной сетью пяти образов [7]. При этом каждому паттерну ставится в соответствие определенный нейрон слоя F2. Тогда входной образ может идентифицироваться как по номеру активного нейрона, так и по его весовым коэффициентам v_{ij}.

ГЛАВА 3. ГИБРИДНЫЕ НЕЙРОННЫЕ СЕТИ

Гибридные нейронные сети представляют собой объединение различного рода нейронных сетей и концепций их обучения. Они предназначены для решения различного рода задач, таких как распознавание образов, прогнозирование, аппроксимация функций и т. д. В данной главе рассматриваются нейронные сети встречного распространения [15, 16], с радиально-базисной функцией активации [17-19], для иерархической классификации образов и для решения задач оптимизации [20, 21]. Наиболее известными среди них являются нейронные сети встречного распространения и с радиально базисной функцией активации. В главе приводится архитектура и алгоритмы обучения гибридных нейронных сетей.

3.1. Нейронные сети встречного распространения

Нейронные сети ***встречного распространения*** (Counterpropagation networks) были предложены в 1987 г Хечт-Нильсоном (Hecht-Nielsen) [15]. Они являются дальнейшим расширением нейронных сетей Кохонена и предназначены для аппроксимации функций:

$$f: R^n \rightarrow R^m.$$

В отличие от сети Кохонена, которая разбивает входное n-мерное пространство на разные области, сеть встречного распространения ставит в соответствие каждой области числовое значение аппроксимируемой функции. Она характеризуется сочетанием двух подходов к обучению: с учителем и без учителя.

Существуют различные варианты нейронных сетей встречного

распространения, которые используют разные методы аппроксимации функций.

3.1.1. Линейная аппроксимация функций поверхностями постоянного уровня

Архитектура нейронной сети встречного распространения изображена на рис. 3.1.

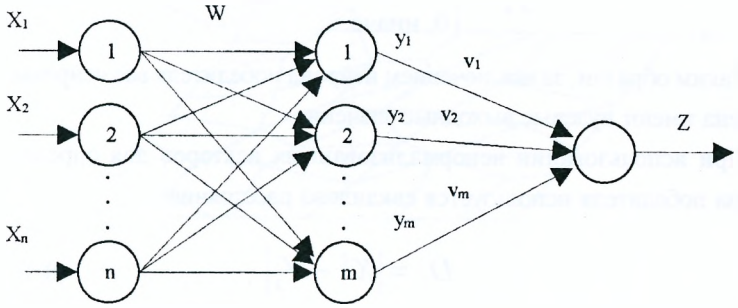


Рис. 3.1. Нейронная сеть встречного распространения

Она состоит из трех слоев, причем входной слой распределяет компоненты входного n-мерного вектора X на все нейроны промежуточного слоя. Промежуточный слой состоит из нейронов Кохонена и работает по принципу "победитель берет всё". В соответствии с ним, если весовые векторы W_j нейронов слоя Кохонена и входные векторы X' являются нормированными, то победителем в конкуренции является нейрон, который имеет максимальное значение скалярного произведения входного вектора на весовой вектор:

$$S_j = \left(W_j, X^l\right) = \sum_i^m \omega_{ij}\, x_i^l \,, \tag{8.1}$$

где w_{ij} - весовой коэффициент между i-тым входным и j-тым нейроном промежуточного слоя, x_i^l - i-тая компонента l-того входного вектора, m - размерность промежуточного слоя.

Тогда выходное значение нейронов слоя Кохонена

$$y_k = \begin{cases} 1, \text{если } \; y_k = \max\limits_j \left\{S_j\right\} \\ 0, \text{иначе.} \end{cases} \tag{8.2}$$

Таким образом, за исключением нейрона победителя все нейроны слоя Кохонена имеют нулевые выходные значения.

При использовании ненормализованных векторов для определения нейрона победителя используется евклидово расстояние:

$$D_j = \left| X^l - W_j \right| . \tag{8.3}$$

В соответствии с этим

$$y_k = \begin{cases} 1, \text{если } D_k = \min\limits_j \left| X^l - W_j \right| \\ 0, \text{иначе.} \end{cases} \tag{8.4}$$

Выходной нейрон сети характеризуется линейной функцией активации. Его выходное значение определяется следующим образом:

$$z = \sum_{i=1}^m v_i y_i \,, \tag{8.5}$$

где v_i - весовой коэффициент i-того входа линейного нейрона.

Если нейрон победитель в слое Кохонена имеет номер k, то выходное значение сети

$$z = v_k . \qquad (3.6)$$

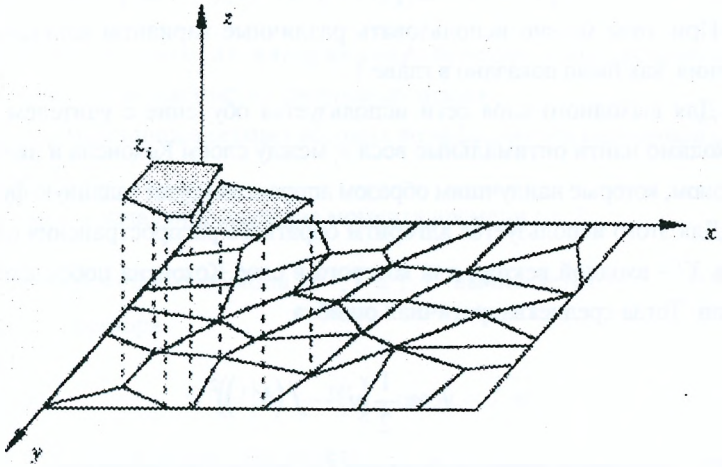


Рис. 3.2. Аппроксимация функции поверхностями постоянного уровня

Таким образом нейронная сеть встречного распространения каждому i-тому кластеру нейрона промежуточного слоя ставит в соответствие весовой коэффициент v_i выходного слоя. Это может использоваться для аппроксимации функции [21], как показано на рис. 3.2.

Здесь карта, изображенная на плоскости (X Y), соответствует разбиению входного пространства на различные области слоем Кохонена. Участок z_i с высотой v_i является аппроксимацией функции в этой области.

Обучение нейронной сети встречного распространения состоит из двух этапов, которые можно осуществить как параллельно, так и последовательно. Первый этап соотвстствует обучению нейронов слоя Кохонена, а второй этап

соответствует обучению выходного слоя. Для обучения слоя Кохонена используется конкурентное обучение (глава 1), где весовые коэффициенты нейрона победителя модифицируются следующим образом:

$$\omega_{ik}(t+1) = \omega_{ik}(t) + \gamma(t) \cdot \left(x_i(t) - \omega_{ik}(t)\right). \quad (3.7)$$

При этом можно использовать различные варианты конкурентного обучения, как было показано в главе 1.

Для выходного слоя сети используется обучение с учителем. Здесь необходимо найти оптимальные веса v_i между слоем Кохонена и линейным нейроном, которые наилучшим образом аппроксимируют заданную функцию $f(X)$. Для этого используется алгоритм обратного распространения ошибки. Пусть X^i - входной вектор, для которого в слое Кохонена побеждает i-тый нейрон. Тогда среднеквадратичная ошибка

$$E = \frac{1}{2}\left(v_i - f\left(X^i\right)\right)^2, \quad (3.8)$$

где $f(X^i)$ - эталонное значение функции для данного входного вектора.

Для минимизации среднеквадратичной ошибки используем градиентный метод в пространстве весовых коэффициентов V. Тогда градиент E определяется, как

$$\frac{\partial E}{\partial v_i} = v_i - f\left(X^i\right). \quad (3.9)$$

В соответствии с этим весовые коэффициенты выходного слоя определяются следующим образом:

$$v_i(t+1) = v_i(t) - \alpha \cdot \left(v_i(t) - f\left(X^i\right)\right), \quad (3.10)$$

где $0 < a < 1$ - скорость обучения.

Следует отметить, что i-тый нейрон может становиться победителем

для целого ряда векторов X, которые отображаются в i-тую область карты Кохонена. Это зависит от размеров кластера, который определяется нейроном победителем. После большого количества итераций вес v_i будет характеризовать среднее значение аппроксимируемой функции $f(X)$ в этой области.

Рассмотрим один из алгоритмов обучения сети встречного распространения. Он состоит из следующих шагов:

1. Случайная инициализация весовых коэффициентов нейронной сети в диапазоне [0, 1].

2. Задается начальное значение момента времени $t = 1$.

3. Для всех входных образов $X^l, l = \overline{1, L}$ вычисляется

а) Норма вектора

$$D^l_j = \left| X^l - W_j \right| , \text{ где } j = \overline{1,m}.$$

б) определяется нейрон победитель

$$D^l_k = \min_j \left\{ D^l_j \right\};$$

с) производится модификация весовых коэффициентов нейрона победителя

$$\omega_{ik}(t+1) = \omega_{ik}(t) + \gamma(t) \cdot \left(X^l_i - \omega_{ij}(t) \right), \forall j = k$$

$$\omega_{ik}(t+1) = \omega_{ik}(t), \forall j \neq k ,$$

где $\gamma(t) = \frac{1}{t}$, $i = \overline{1,n}$, $j = \overline{1,m}$;

d) в соответствии с эталонным значением функции $f(X^l)$ осуществляется модификация k-го весового коэффициента последнего слоя

$$v_k(t+1) = v_k(t) - \alpha \cdot \left(v_k(t) - f\left(X^1\right)\right) \cdot$$

4. Увеличиваем значение времени и процесс повторяется, начиная иная с п. 3.

Обучение производится до желаемой точности аппроксимации функции и в общем случае требует количества итераций $t \geq 500$. Данная сеть аппроксимирует значения функции, которые принадлежат одному кластеру, поверхностями постоянного уровня.

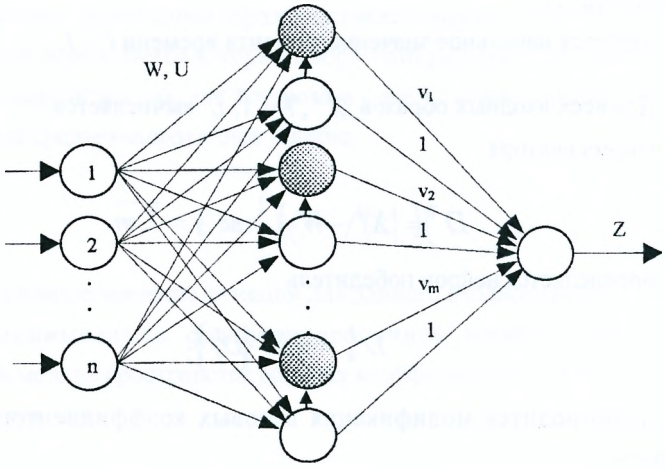


Рис. 3.3. Модифицированная нейронная сеть встречного распространения

3.1.2. Линейная аппроксимация функций поверхностями переменного уровня

Для более точной аппроксимации функции *f(x)* может использоваться модифицированная нейронная сеть встречного распространения [14]. Архитектура ее изображена на рис. 3.3 .

В отличие от стандартной сети встречного распространения, здесь промежуточный слой состоит из нейронов Кохонена (не заштрихованные кружки) и линейных нейронов (заштрихованные кружки), количество которых равняется между собой. Такие нейроны образуют пары, причем каждый нейрон Кохонена в паре имеет горизонтальную связь с соответствующим ему линейным нейроном, как показано на рис.3.3. Кроме этого все нейроны промежуточного слоя соединены с выходным нейроном, который имеет линейную функцию активации. Будем использовать последовательную нумерацию пар нейронных элементов. Обозначим через w_{ij} и u_{ij} соответственно весовые коэффициенты нейронов Кохонена и линейных нейронов промежуточного слоя. Тогда выходное значение p_j j-того линейного нейрона можно определить так

$$p_j = y_j \cdot \sum_{i=1}^{n} u_{ij} x_i , \qquad (3.11)$$

где y_j - выходное значение нейрона Кохонена в j-той паре нейронов.

Выходное значение j-того нейрона Кохонена равняется единице, если этот нейрон является победителем в конкурентной борьбе, и нулю в противном случае. Выходное значение нейронной сети будет равно

$$z = \sum_{j=1}^{m} \left(v_j y_j + p_j \right) = \sum_{j=1}^{m} y_j \cdot \left(v_j + \sum_{i=1}^{n} u_{ij} x_i \right). \qquad (3.12)$$

Весовые коэффициенты от линейных нейронов промежуточного слоя к выходному нейрону равняются единице.

Если нейрон победитель в промежуточном слое имеет номер k, то

$$z = v_k + p_k = v_k + \sum_{i=1}^{n} u_{ik} x_i . \qquad (3.13)$$

Отсюда следует, что в отличие от предыдущего варианта здесь входной вектор отображается на поверхность переменного уровня, высота которой

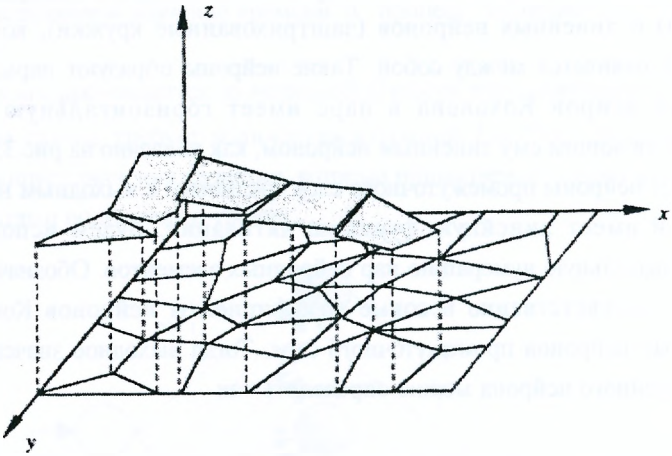


Рис. 3.4. Аппроксимация функции поверхностями переменного уровня

изменяется в пределах кластера по линейному закону. Поэтому векторы, характеризующие один и тот же кластер, будут иметь различные выходные значения.

Пример аппроксимации функции при помощи нейронной сети встречного распространения изображен на рис. 3.4.

Обучение здесь, как и в предыдущем случае осуществляется в два этапа, которые могут проходить параллельно. На первом этапе происходит обучение нейронов слоя Кохонена, а на втором - линейных нейронов промежуточного слоя и выходного нейрона. Градиенты весовых коэффициентов при этом определяются, как

$$\frac{\partial E}{\partial v_k} = \frac{\partial E}{\partial z}\frac{\partial z}{\partial v_k} = \left(z - f(X)\right), \qquad (3.14)$$

$$\frac{\partial E}{\partial u_{ik}} = \frac{\partial E}{\partial z}\frac{\partial z}{\partial u_{ik}} = \big(z - f(X)\big)x_i. \qquad (3.15)$$

Тогда модификация весовых коэффициентов линейных нейронов происходит следующим образом:

$$v_k(t+1) = v_k(t) - \alpha(z - f(X)), \qquad (3.16)$$

$$u_{ik}(t+1) = u_{ik}(t) - \alpha(z - f(X))x_i. \qquad (3.17)$$

На выходе нейронных сетей встречного распространения может применяться не один, а несколько линейных нейронов. В этом случае будет происходить отображение входного вектора на более сложную поверхность, которая имеет, например, вид ломаной.

3.1.3. Нелинейная аппроксимация функций

Для нелинейной аппроксимации функций можно использовать вместо линейных нейронов промежуточного слоя (рис. 3.3) нейронные элементы с нелинейной функцией активации. Пусть F – функция активации нейронных элементов промежуточного слоя. Тогда выходное значение *k*-го нейрона можно представить в следующем виде:

$$p_k = F(S_k) = F\Big(\sum_{i=1}^{n} u_{ik}x_i\Big). \qquad (3.18)$$

Выходное значение нейронной сети определяется, как

$$z = v_k + p_k \qquad (3.19)$$

В соответствии с алгоритмом обратного распространения ошибки модификация весовых коэффициентов выходного слоя будет определяться, как

$$v_i(t+1) = v_i(t) - \alpha(z - f(X)), \tag{3.20}$$

а для нейронных элементов промежуточного слоя

$$u_{ik}(t+1) = u_{ik}(t) - \alpha(z - f(X))\frac{\partial F}{\partial S_k}x_i \tag{3.21}$$

Если в качестве функции активации применяется сигмоидная функция, то

$$\frac{\partial F}{\partial S_k} = p_k(1 - p_k), \tag{3.22}$$

тогда

$$u_{ik}(t+1) = u_{ik}(t) - \alpha(z - f(X))p_k(1 - p_k)x_i. \tag{3.23}$$

Применяя рассмотренные выше правила обучения, можно настроить нейронную сеть для нелинейной аппроксимации функций.

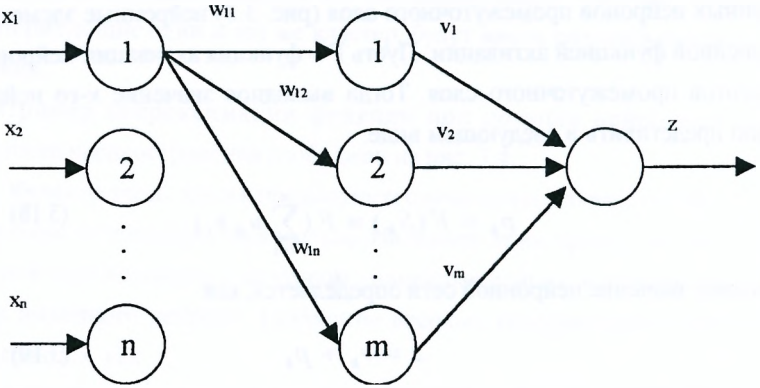


Рис. 3.5. Архитектура RBF сети

3.2. Нейронные сети с радиально-базисной функцией

Нейронные сети с ***радиально-базисной функцией*** (Radial Basis Function Network) являются дальнейшим развитием сетей встречного распространения. Они были предложены в 1989 году [17] и предназначены для решения задач распознавания образов, прогнозирования, сжатия данных и аппроксимации функций [18, 19]. Архитектура RBF нейронной сети изображена на рис. 3.5.

Она состоит из трех слоев. Нейроны входного слоя выполняют распределительные функции. Промежуточный слой состоит из нейронов Кохонена, а в качестве выходного слоя используются нейронные элементы с линейной функцией активации. Недостаток сети Кохонена состоит в том, что в процессе функционирования для нее необходима глобальная информация о состоянии нейронных элементов. При этом слой Кохонена всегда должен определять, какой из нейронов обладает максимальной активностью и присваивать ему единичное значение, а остальным нейронам нулевые значения. В отличии от этого, в RBF сети нейроны слоя Кохонена функционируют с реальными выходными значениями, которые определяются функцией Гаусса с нормальным законом распределения. В соответствии с этим выходное значение j-го нейрона слоя Кохонена определяется следующим образом:

$$y_j = \exp\left(-\frac{\left|X - W_j\right|^2}{2\sigma_j^2}\right), \qquad (3.24)$$

где X - входной вектор, $W_j = \{w_{1j}, w_{2j}..w_{mj}\}$ - вектор весов j-го нейрона слоя Кохонена, σ_j - дисперсия, которая характеризует ширину радиально-

базисной функции.

Так как

$$D_j = \left|X - W_j\right|^2 = \sum_{i=1}^{n} \left(x_i - w_{ij}\right)^2, \qquad (3.25)$$

то

$$y_j = \exp\left(-\frac{D_j}{2\sigma_j^2}\right). \qquad (3.26)$$

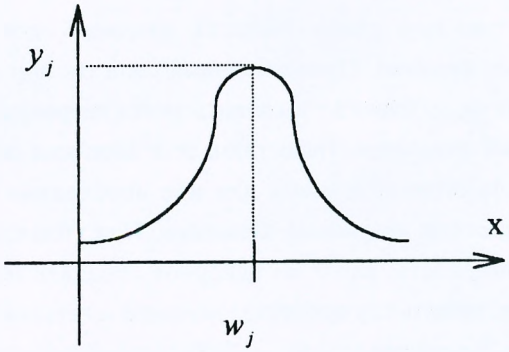


Рис. 3.6. Функция Гаусса

где D_j - характеризует евклидово расстояние между весовым вектором j-го нейрона W_j и входным вектором X.

Функция Гаусса для j-го нейрона слоя Кохонена изображена на рис. 8.6.

Ширина радиально базисной функции σ_j характеризует среднее расстояние между центром кластера, который образуется j-м нейронным

элементом и его ближайшими соседями:

$$\sigma_j^2 = \frac{1}{p}\sum_{k=1}^{p} \left| W_j - W_k \right|^2 , \qquad (3.27)$$

где p - количество ближайших соседей j-го нейронного элемента.

В случае линейной организации слоя Кохонена p=2, а при матричной организации обычно p = 4.

Расстояние между центрами кластеров, образуемых j-м и k-м нейронными элементами определяется обычно как евклидово расстояние

$$\left| W_j - W_k \right|^2 = \sum_{i=1}^{n} \left(w_{ij} - w_{ik} \right)^2 \qquad (3.28)$$

Тогда

$$\sigma_j^2 = \frac{1}{p}\sum_{k=1}^{p}\sum_{i=1}^{n} \left(w_{ij} - w_{ik} \right)^2 \qquad (3.29)$$

Выходное значение z RBF сети определяется как взвешенная сумма выходных значений слоя Кохонена:

$$z = \sum_{j=1}^{m} v_j y_j = |V||Y| \cos \beta , \qquad (3.30)$$

где v_j - весовой коэффициент j-го входа линейного нейрона, $\beta = \hat{V,Y}$ – угол между векторами V и Y.

Согласно выражению (3.30) выходное значение сети z равнятся проекции вектора $Y = (y_1, y_2 .. y_m)$ на вектор связей $V = (v_1, v_2 .. v_m)$. Так как длина вектора $|Y| = VAR$, то выходное значение z может неадекватно отражать желаемые значения сети. Поэтому целесообразно нормировать выходные

значения слоя Кохонена так, чтобы

$$\sum_{j=1}^{m} y_j = 1 \qquad (3.31)$$

Тогда функция передачи j-го нейрона будет определяться в соответствии с нормированной функцией Гаусса [14]:

_____ (3.32)

Согласно выражению (3.32) для вычисления необходимо, чтобы значения $\exp\left(-|X - W_k|^2 / 2\sigma_k^2\right)$ от других нейронов передавались j-му нейронному элементу. Это предполагает вертикальные связи между нейронами слоя Кохонена.

Если входные векторы являются нормированными, т. е. их концы находятся на поверхности гиперсферы радиусом R, то

$$|X| = \sqrt{\sum_{i=1}^{n} x_i^2} = R \cdot \qquad (3.33)$$

Тогда вместо евклидовой меры можно использовать взвешенную сумму:

$$I_j = \sum_{i=1}^{n} w_{ij} x_i \qquad (3.34)$$

В этом случае функция активации j-го нейрона слоя Кохонена равняется

$$y_j = \exp\left(\frac{I_j - R^2}{\sigma_j^2}\right). \qquad (3.35)$$

Таким образом, в соответствии с выражениями (3.26), (3.32), (3.35) в RBF сети можно использовать различные функции активации нейронных элементов.

Обучение нейронной сети с радиально-базисной функцией активации нейронных элементов производится последовательно в два этапа. На первом этап производится обучение слоя Кохонена в соответствии с правилом «победитель берет все». Для этого используется один из вариантов конкурентного метода, которые были рассмотрены в главе 1. В результате этого входные образы группируются в кластеры и определяется в соответствии с выражением (3.29) ширина радиально-базисной функции для каждого нейрона слоя Кохонена. На втором этапе используется обучение с учителем. При этом весовые коэффициенты v_j инициализируются вначале случайным образом, входные образы подаются на сеть и происходит настройка синаптических связей V в соответствии с алгоритмом обратного распространения ошибки. Для этого необходимо минимизировать среднеквадратичную ошибку сети.

$$E = \frac{1}{2}(z - f(X))^2, \tag{3.36}$$

где $f(X)$ - эталонные значения функции для данного вектора X.

Используя метод градиентного спуска

$$v_j(t+1) = v_j(t) - \alpha \frac{dE}{dv_j(t)} \tag{3.37}$$

Отсюда

$$\frac{dE}{dv_j(t)} = \frac{dE}{dz}\frac{dz}{dv_j(t)} = y_j(z - f(X)) \tag{3.38}$$

Тогда

$$v_j(t+1) = v_j(t) - \alpha\, y_j(z - f(X)), \qquad (3.39)$$

где y_j определяется в соответствии с одной из функций активации нейронных

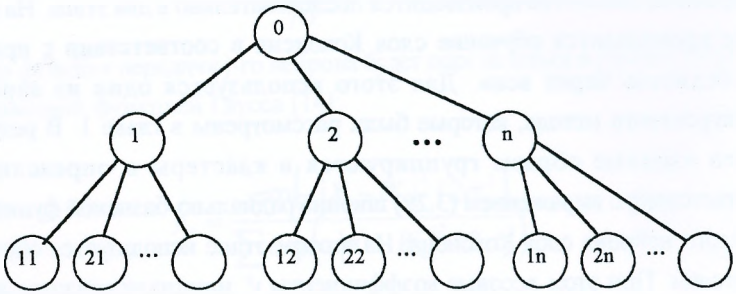


Рис. 3.7 Иерархическая классификация образов; О – входное пространство образов; j – номер класса; (ij) – характеризует i-й образ принадлежащий j-му классу.

элементов, рассмотренных выше.

Обучение на втором этапе производиться до тех пор, пока суммарная среднеквадратичная ошибка сети не станет меньше заданной.

3.3. Иерархический классификатор

В данном разделе описывается нейронная сеть для иерархической классификации входного пространства образов. Такая сеть осуществляет разбиение пространства входных образов на классы, где каждому классу поставлен в соответствие определенный набор образов. Это схематично представлено на рис. 3.7. в виде дерева.

В соответствии с этим, при функционировании нейронной сети

Рис.3.8. Общая структура иерархического классификатора

определяется не только образ, но и номер класса к которому он принадлежит.

3.3.1. Архитектура нейронной сети

Иерархический классификатор представляет собой объединение двух различных нейронных сетей (рис. 3.8). Первая сеть состоит из слоя Кохонена с конкурентным обучением и предназначена для разбиения входного пространства образов на классы

Второй слой представляет собой совокупность ассоциативных нейронных

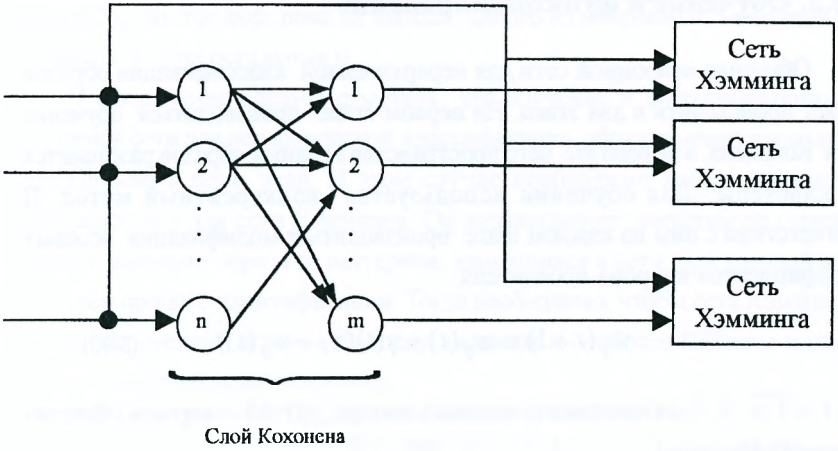


Рис. 3.9. Архитектура иерархического классификатора

сетей, которые хранят образы, принадлежащие различным классам. Количество ассоциативных нейронных сетей ровняется количеству классов. В качестве ассоциативной нейронной сети может использоваться сеть Хэмминга или Хопфилда. Выходным сигналом слоя Кохонена является позиционный код, который соответствует номеру класса и возбуждает определенную ассоциативную сеть второго слоя. В результате этого входной образ поступает на соответствующую ассоциативную сеть. Если входной образ не принадлежит к уже имеющимся классам, то для него может резервироваться новый класс и соответствующая ему ассоциативная сеть.

Пусть в качестве ассоциативной памяти используется нейронная сеть Хэмминга. Для этого случая архитектура иерархического классификатора представлена на рис. 3.9.

Такая нейронная сеть обучается без учителя и характеризуется самоорганизацией в процессе работы.

3.3.2. Обучение и функционирование

Обучение нейронной сети для иерархической классификации образов может происходить в два этапа. На первом этапе производиться обучение слоя Кохонена, в результате чего пространство входных образов разбивается на кластеры. Для обучения используется конкурентный метод. В соответствии с ним на каждом шаге производиться модификация весовых коэффициентов нейрона победителя:

$$w_{ij}(t+1) = w_{ij}(t) + \gamma(t)(x_i - w_{ij}(t)), \qquad (3.40)$$

где $i = \overline{1,n}$; x_i – i-ая компонента входного вектора; $\gamma(t)=1/t$ – скорость обучения в момент времени t.

На втором этапе обучения производиться настройка весовых коэффициентов слоя Хэмминга согласно выражениям приведенным в разделе

5.

Алгоритм функционирования нейронной сети для иерархической классификации образов состоит из следующих шагов:

1) подается входной образ на нейронную сеть;

2) вычисляется норма вектора

$$D_j = \left| X - W_j \right|,$$

где $j = \overline{1,m}$;

3) определяется нейрон победитель, обеспечивающий минимальное расстояние D_j

$$D_k = \min_j D_j$$

4) в соответствии с номером нейрона-победителя возбуждается одна из нейронных сетей слоя Хэмминга, на которую подается входной образ X;

5) производиться итерационная процедура схождения сети Хэмминга (раздел 5), до тех пор, пока на выходе одного из нейронных элементов не останется 1, а на остальных 0.

В заключение следует отметить, что обучение и функционирование нейронной сети для иерархической классификации образов может проходить параллельно в один этап. В этом случае необходимо ввести параметр бдительности ρ для сети Хэмминга. Он характеризует допустимую степень отличия входного образа от паттернов, хранящихся в сети, при которой еще возможен процесс идентификации. Тогда необходимо, чтобы сеть Хэмминга определяла степень отличия d между входным и наиболее близким к нему образом, который хранится в сети:

$$\delta = \max_j \{n - d_j\}, \qquad (3.41)$$

где d_j – расстояние Хэмминга между входным и j-м образом, хранящимся в

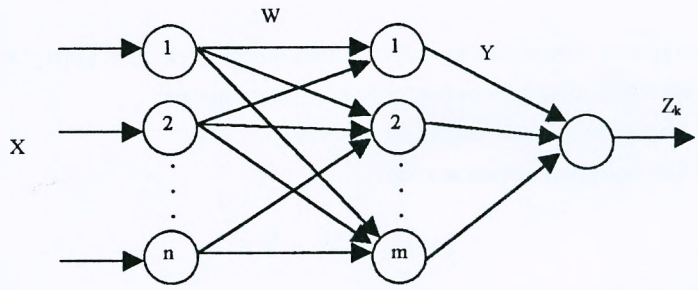


Рис. 3.10. Общая структура нейронной сети для решения задач оптимизации

сети.

Чем меньше δ, тем больше различаются соответствующие образы. Если $\delta < \rho$, то производится обучение сети Хэмминга на текущий входной образ. В противном случае осуществляется идентификация входного образа.

3.4. Решение задач оптимизации

В разделе рассматривается решение задач оптимизации при помощи нейронных сетей с фиксированными связями [20, 21]. Определение весовых коэффициентов здесь происходит исходя из условия задачи и в общем случае представляет собой процесс перебора различных вариантов решения. Обобщенная архитектура нейронной сети для решения задач оптимизации приведена на рис.3.10.

Она представляет собой трехслойную сеть с прямыми связями. Первый слой нейронных элементов является распределительным. Нейроны второго слоя формируют различные варианты решения задачи. Выходной нейронный

элемент предназначен для определения оптимального решения задачи в соответствии с заданным критерием оптимизации.

$$Z_k = \underset{j}{opt}(y_j), \qquad (3.42)$$

где к – номер нейронного элемента, который идентифицирует оптимальный вариант решения; Z_k – значение целевой функции при оптимальном варианте решения задачи. Такие нейронные сети были предложены в работах [20-21].

3.4.1. Задача о кратчайшем пути

Формулировка задачи о кратчайшем пути состоит в следующем. Имеется начальный и конечный пункты движения, между которыми существует множество возможных путей. Известны расстояния между различными пунктами движения. Тогда необходимо определить кратчайший путь между начальной и конечной точками движения. Определим структуру нейронной сети. Количество нейронов входного слоя равняется:

$$p = (n-1)(n-2)+1, \qquad (3.43)$$

где n – общее количество городов или пунктов движения.

На каждый нейрон входного слоя подается расстояние D_{ij} между соответствующими пунктами движения i и j. Причем $i = \overline{1, n-1}$; $j = \overline{1, n}$; $i \neq j$; $j > i$ для $i=1$ или $j=n$.

Так, если n = 4, то p = 7 и входной вектор расстояний определяется следующим образом:

$$D = (D_{12}, D_{13}, D_{14}, D_{23}, D_{24}, D_{32}, D_{34}), \qquad (3.44)$$

Здесь предполагается, что 1 характеризует начальный, а 4 – конечный пункты движения.

Второй слой нейронных элементов формирует возможные пути между

начальным и конечным пунктами движения. Количество нейронов в нем определяется, как

$$m = \sum_{i=1}^{n-1} \binom{n-2}{n-i-1} (n-i-1)! = (n-2)! \sum_{i=1}^{n-1} \frac{1}{(i-1)!}, \quad (3.45)$$

где $0! = 1$.

Так как для больших значений n

$$\sum_{i=1}^{n-1} \frac{1}{(i-1)!} = e,$$

то

$$m \approx e(n-2)!.$$

Для упрощения обозначим компоненты входного вектора в линейной системе координат

$$D = (D_1, D_2, ..., D_p).$$

Тогда выходные значения нейронных элементов промежуточного слоя определяются следующим образом:

$$y_j = \sum_{i=1}^{p} w_{ij} D_i, \quad (3.46)$$

где $j = \overline{1,m}$.

Весовые коэффициенты нейронной сети формируются таким образом, чтобы получить совокупность возможных маршрутов. Размерность матрицы весовых коэффициентов равняется pxm. Так, если n=4 то m=5 и матрица синаптических связей имеет следующий вид:

$$W = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Она является бинарной, если возможны любые пути между начальным и конечным пунктами движения. Если между какими-нибудь пунктами движения не существует пути, то вместо соответствующего единичного значения необходимо поставить бесконечность (∞). Так, если нет пути между

Таблица 3.1

	1	2	3	4
1	0	25	100	75
2	25	0	55	35
3	100	55	0	15
4	75	35	15	0

первым и вторым пунктами маршрута, т.е. $D_{12}=\varnothing$, то $w_{11}=\infty$. Каждый столбец матрицы W характеризует весовые коэффициенты соответствующего нейронного элемента. Нейронный элемент третьего слоя определяет кратчайший путь

$$Z_k = \min_j \{y_j\}, \qquad (3.47)$$

где Z_k – длина оптимального маршрута; к – номер маршрута.

По номеру нейронного элемента-победителя k можно определить кратчайший маршрут:

$$D_o = D^T W_k^T, \qquad (3.48)$$

где W_k – весовой вектор k-го нейрона.

Пример 8.1. Пусть дана матрица расстояний между городами (табл.3.1). Необходимо определить кратчайший путь между городами 1 и 4.

Входной вектор расстояний определяется как:

$$D = (25,100,75,55,35,55,15)$$

Каждый нейрон промежуточного слоя определяет один из возможных маршрутов. Тогда

$$y_1 = 25+55+15 = 95$$

$$y_2 = 25+35 = 60$$

$$y_3 = 100+35+55=190$$

$$y_4 = 100+15=115$$

$$y_5 = 75$$

Выходное значение сети определяется, как

$$Z_k = \min_j \{y_j\} = 60,$$

$$k = 2 \cdot$$

Весовые коэффициенты второго нейрона промежуточного слоя соответствуют маршруту

$$D_o = \begin{bmatrix} D_{12} \\ D_{13} \\ D_{14} \\ D_{23} \\ D_{24} \\ D_{32} \\ D_{34} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} D_{12} \\ D_{24} \end{bmatrix}$$

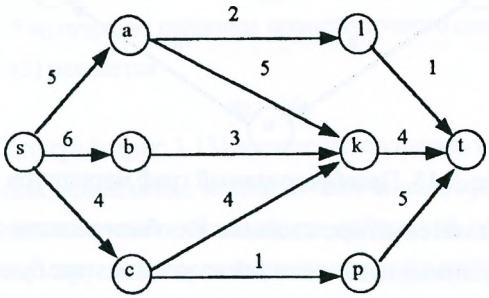


Рис. 3.12. Исходный граф маршрутов

В результате получим следующий маршрут:

$$1 \rightarrow 2 \rightarrow 4 .$$

Рассмотренная нейронная сеть характеризуется значительной сложностью при большой размерности задачи. Поэтому ее можно использовать на заключительных этапах решения задачи для поиска оптимального варианта. Другим способом является адаптация нейронной сети к алгоритму оптимизации, применяемому для решения задачи. Так, например, если для определения кратчайшего пути используется динамическое программирование, то легко снизить размерность нейронной

сети.

Рассмотрим общий подход применения динамического программирования и нейронной сети для нахождения кратчайшего пути.

Пусть возможные маршруты между начальной точкой s и конечной t, представлены в виде графа (рис.3.12).

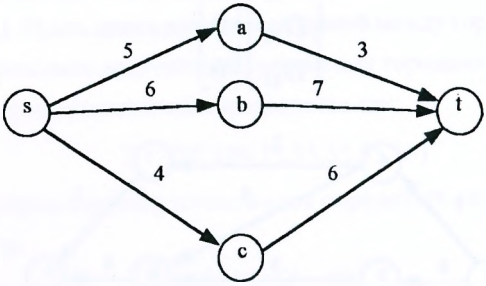


Рис. 3.13. Преобразованный граф маршрутов

Такой граф является четырехслойным. Разобьем решение задачи на этапы и для определения оптимальных решений на каждом этапе будем использовать нейронную сеть. Первый этап решения охватывает три последних слоя графа. Здесь необходимо найти оптимальные пути из пунктов a, b и c в пункт t. Для каждой из этих задач формируем нейронную сеть. Так, общее количество городов при движении из a в t равняется 4. Поэтому структура сети будет такой же, как и в примере 3.1. В результате выполнения первого этапа, получим

$$Z_2(a) = 3;\ \ a \to l \to t,$$

$$Z_1(b) = 7;\ \ b \to k \to t,$$

$$Z_4(c) = 6;\ \ c \to p \to t.$$

Представим теперь граф в следующем виде:

Для нахождения оптимального маршрута из s в t будем использовать

нейронную сеть для n = 5.

Тогда

$$Z_7(S) = 8,$$

что соответствует маршруту

$$s \to a \to l \to t.$$

Если учитывать при построении нейронной сети только реальные маршруты, то количество нейронов промежуточного слоя значительно уменьшается.

Так, для n = 5 количество нейронов промежуточного слоя в соответствии с выражением (3.45) равняется

$$m = 16.$$

Исходя из реального графа (рис.3.13) достаточно положить m = 3.

Таким образом, применение метода динамического программирования позволяет значительно снизить размерность нейронной сети. Аналогично можно адаптировать другие алгоритмы оптимизации к нейронной сети.

3.4.2. Задача о рюкзаке

Задача о рюкзаке (knapsack problem) является NP-полной и формулируется следующим образом [22-23]. Имеется совокупность предметов $U=(u_1,u_2,\ldots,u_n)$ и для каждого из них известен объем $V(u_i)$ и стоимость $C(u_i)$, где $i = \overline{1,n}$. Требуется заполнить предметами рюкзак ограниченного объема T таким образом, чтобы максимизировать стоимость упакованных вещей или сделать ее больше заданной K. Тогда математическую формулировку задачи о рюкзаке можно представить для двух случаев следующим образом:

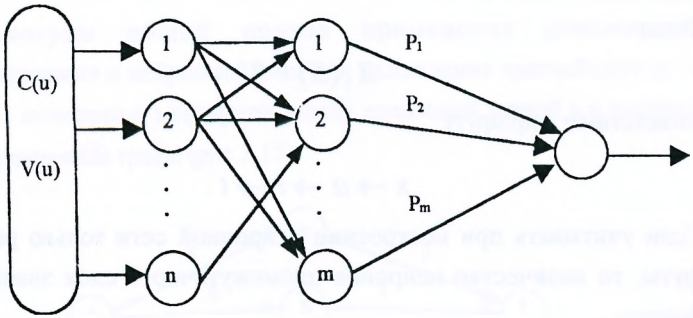


Рис. 3.14. Нейронная сеть для решения задачи о рюкзаке

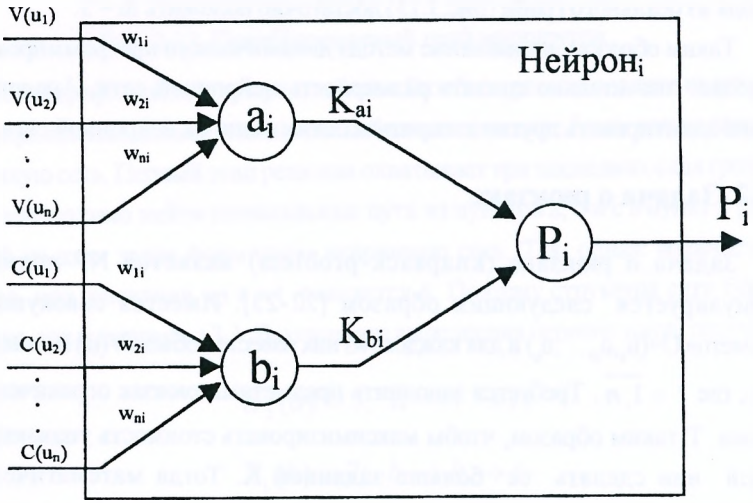


Рис. 3.15. Схема нейронного элемента второго слоя

$$\sum_{u} V(u) \le T \quad \text{и} \quad \max_{u}\left\{\sum_{u} C(u)\right\}, \qquad (3.49)$$

$$\sum_{u} V(u) \le T \quad \text{и} \quad \sum_{u} C(u) \ge K, \qquad (3.50)$$

где K – значение заданной стоимости. Общая архитектура нейронной сети для решения задачи о рюкзаке представлена на рис. 3.14.

Она состоит из трех слоев нейронных элементов. Первый слой выполняет распределительные функции. На вход i-го нейрона первого слоя поступает стоимость $C(u_i)$ и объем $V(u_i)$ i-го предмета, которые он распределяет на каждый из нейронов второго слоя. Второй слой предназначен для определения возможных вариантов решения задачи. Каждый нейрон второго слоя состоит из трех элементарных нейронов (рис. 3.15), которые выполняют различные функции.

Элемент b_i генерирует i-й вариант решения задачи. Для этого он вычисляет следующую функцию:

$$K_{bi} = \sum_{j=1}^{n} w_{ji} C(u_j) \qquad (3.51)$$

Элемент a_i анализирует i-й вариант на принадлежность его допустимому объему T. Он является нейронным элементом с пороговой функцией активации и выполняет следующие действия:

$$S_i = \sum_{j=1}^{n} w_{ji} V(u_j) - T, \qquad (3.52)$$

$$K_i = \begin{cases} 1, & если \quad S_i \le 0 \\ 0, & иначе \end{cases} \qquad (3.53)$$

Элемент P_i производит анализ возможности участия i-го нейронного элемента в дальнейшей конкурентной борьбе. Для этого он определяет

$$P_i = \begin{cases} K_{bi}, если \quad K_{ai} = 1 \\ 0, \quad иначе \end{cases} \qquad (3.54)$$

Таким образом i-й нейронный элемент промежуточного слоя формирует соответствующий вариант решения задачи K_{bi}, если он не превышает допустимый объем T.

Третий слой нейронной сети состоит из одного элемента, который определяет оптимальный вариант решения задачи:

$$Z_k = \max_i \{P_i\}, \qquad (3.55)$$

где Z_k – максимальная стоимость предметов в рюкзаке; k – номер нейронного элемента, который идентифицирует оптимальный вариант решения задачи. По номеру k нейронного элемента победителя можно определить решение задачи:

Количество нейронных элементов второго слоя может быть различным в зависимости от технологии решения задачи о рюкзаке. В общем случае, при анализе всех вариантов решения скрытый слой содержит следующее количество нейронных элементов:

$$m = 2^n - 1. \qquad (3.57)$$

Тогда весовой вектор i-го нейрона соответствует двоичной записи числа i. Так, если i=5 и n=6, то

$$W_5=(000101) \qquad (3.58)$$

Матрица весовых коэффициентов формируется таким образом, что каждый ее столбец характеризует весовые коэффициенты соответствующего нейронного элемента. Так для n = 3 и m = 7 весовая матрица имеет следующий вид:

$$W=\begin{bmatrix}0&0&0&1&1&1&1\\0&1&1&0&0&1&1\\1&0&1&0&1&0&1\end{bmatrix} \qquad (3.59)$$

Недостатком описанной выше сети является большое количество нейронов скрытого слоя. Для устранения этого можно использовать нейронную сеть на отдельных этапах решения задачи или адаптировать оптимизационный алгоритм к нейронной сети, как было показано в предыдущем разделе. Другим вариантом является уменьшение количества нейронов скрытого слоя за счет увеличения числа тактов работы сети. Предположим, что имеется нейронная сеть для решения задачи о рюкзаке, которая имеет размерность n. Необходимо решить на такой сети задачу, размерность которой равняется r, где r>n. Обозначим

$$m = r - n \qquad (3.60)$$

и рассмотрим нейронную сеть, которая имеет (n + m) входных и 2^n -1 скрытых нейронных элементов. Для решения на такой сети задачи размерности r необходимо затратить 2^m тактов работы, причем весовые коэффициенты нейронных элементов в каждый такт функционирования будут изменяться в соответствии с матрицей весовых коэффициентов.

Пример 3.2. Рассмотрим пример решения нейронной сетью задачи о рюкзаке размерности n = 3. Пусть

$$V(k)=\{5,7,10\},$$

$$C(k)=\{3,2,1\},$$

$$T=17.$$

Количество нейронов скрытого слоя ровняется

$$m = 2^3 - 1 = 7$$

Матрица весовых коэффициентов нейронной сети определяется в соответствии с (3.59). Тогда результаты вычислений нейронных элементов скрытого слоя можно представить в виде таблицы 3.2.

Определим выходное значение нейронной сети:

$$Z_k = \max_i \{P_i\} = 5,$$

$$k = 6.$$

В соответствии с номером k нейрона и его весовым вектором W_k идентифицируем решение задачи о рюкзаке:

$$U = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

Таким образом оптимальное решение достигается при упаковке в рюкзак предметов u_1 и u_2.

Таблица 3.2

№	K_a	K_b	P
1	1	1	1
2	1	2	2
3	1	3	3
4	1	3	3
5	1	4	4
6	1	5	5
7	0	6	0

ГЛАВА 4. ПРИМЕНЕНИЕ НЕЙРОННЫХ СЕТЕЙ ДЛЯ УПРАВЛЕНИЯ

В настоящее время исследования в области искусственных нейронных сетей ориентированы в основном на создание специализированных систем для решения конкретных задач. Одной из областей, где нейронные сети нашли широкое применение являются робототехника и различного рода системы для управления транспортными средствами. В отличии от традиционных методов нейросетевой аппарат создает потенциальные предпосылки для создания самообучающихся и самоорганизующихся систем.

В главе рассматривается применение нейронных сетей для автономного управления транспортными средствами. В качестве транспортных средств могут использоваться мобильные роботы или автомобиль, которые оснащены сенсорными устройствами для отображения окружающей обстановки. После обучения нейронная сеть на основе информации от сенсорных устройств должна обеспечивать корректное управление движением. Возможность создания таких систем базируется на обобщающей способности нейронных сетей, которая позволяет интегрировать частные данные для определения закономерностей процесса. В результате этого нейронная сеть способна выдавать правильную реакцию на входных данных, которые не входили в обучающую выборку. Общая модель взаимодействия таких систем с внешней средой изображена на рис. 4.1.

Самоорганизация здесь происходит в процессе обучения с целью адаптации к внешней среде. Данная схема эквивалентна модели взаимодействия индивидуума с внешней средой, которая была рассмотрена в главе 1 (книга 1).

Рис. 4.1. Взаимодействие нейронной системы с внешней средой

Во многих лабораториях мира разрабатываются системы для управления роботами. Существует большое количество моделей роботов [24-29] и различных проектов, вплоть до создания человекоподобных роботов. На рис.4.2 – 4.4 изображены различные модели роботов [24-26].

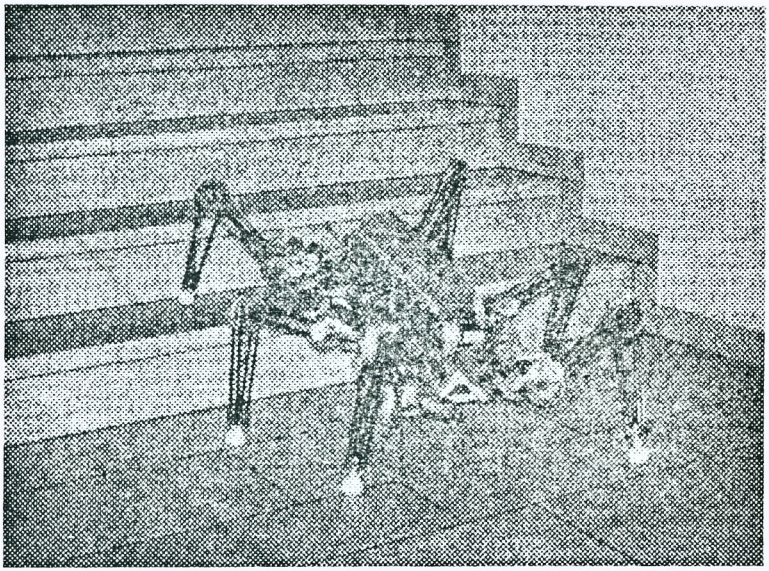


Рис. 4.2. Модель робота "Lauron"

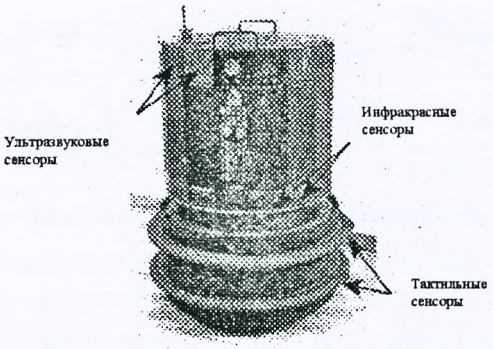


Рис. 4.3. Мобильный робот "Nomad 200"

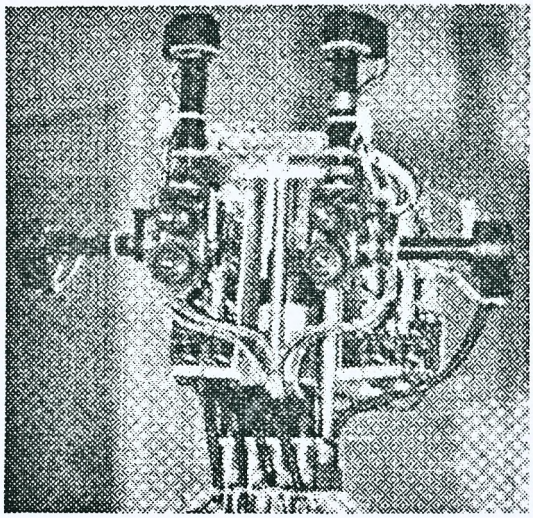


Рис. 4.4. Голова робота, разрабатываемого в лаборатории искусственного интеллекта Масачутесского технологического института

4.1. Управление движением робота по заданной траектории

Одной из задач возникающей при управлении мобильным роботом является движение его по заранее заданной траектории. При этом траектория может задаваться в виде разметки на дороге или в цеху. Задача робота состоит в том, чтобы двигаясь по известной траектории, достичь конечной точки движения. Такой подход позволяет создавать относительно дешевые автономные системы для перевозки грузов на предприятии. В этом случае к ведущему роботу, который отслеживает траекторию движения, могут подсоединяться ведомые роботы, которые образуют автопоезд.

4.1.1. Общая структура системы

Общая структура системы автономного управления мобильным роботом приведена на рис. 4.5.

Мобильный робот оборудован видеокамерой, которая предназначена для отображения заданной траектории. Блок обработки видеоизображений преобразует изображение от видеокамеры в бинарную матрицу, как показано на рис. 4.6.

Размерность матрицы может быть различной. Для моделирования

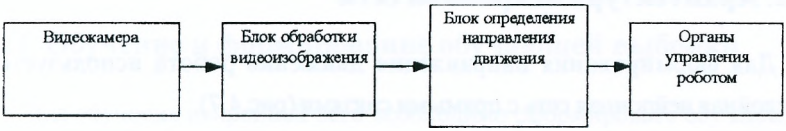
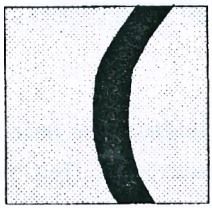
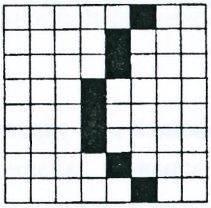


Рис. 4.5. Структура системы автономного управления мобильным роботом



а



б

$$\begin{bmatrix} 0&0&0&0&0&1&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0 \end{bmatrix}$$

в

Рис 4.6. Этапы преобразования изображения траектории от видеокамеры: а – исходное видеоизображение траектории; б – проекция изображения на матрицу размерностью 8x8; в – бинарный массив.

такой системы здесь использовалась бинарная матрица размером 8x8.

Блок определения направления движения представляет собой многослойную нейронную сеть с прямыми связями. Он формирует в каждый момент времени направление движения робота, которое подается в органы управления. Задача такой системы состоит в обеспечении устойчивого управления роботом при движении по различным траекториям. При этом робот должен корректно проходить участки траектории, с формой которых он не был знаком на этапе обучения.

4.1.2. Архитектура нейронной сети

Для формирования направления движения робота используется трехслойная нейронная сеть с прямыми связями (рис.4.7).

Входной слой состоит из 64 нейронных элементов, на который подается бинарная матрица изображения траектории. Промежуточный слой состоит из 15, а выходной слой из 19 нейронных элементов. Каждому входному

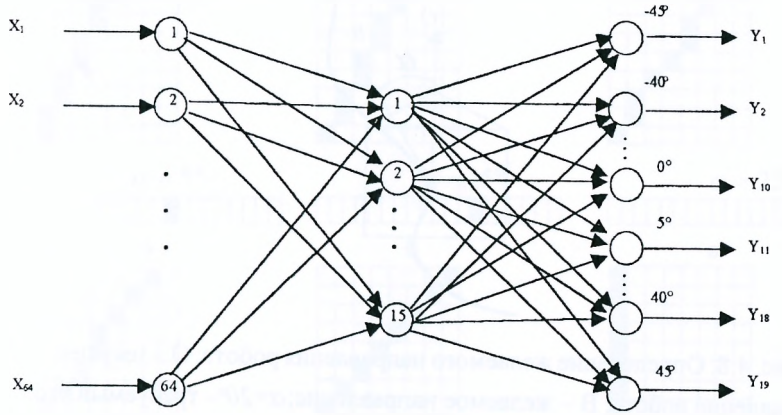


Рис. 4.7. Архитектура нейронной сети: $x_1..x_{64}$ – элементы бинарного массива; $y_1..y_{19}$ – результирующий вектор направления движения

нейрону сети поставлено в соответствие определенное направление движения из диапазона [-45°, 45°]. Шаг дискретизации составляет при этом 5°. В каждый момент времени на выходе нейронной сети является активным только один нейронный элемент, который определяет текущее направление движения робота. В качестве функции нелинейного преобразования используется сигмоидная функция активации.

4.1.3. Обучение и формирование обучающей выборки

Для обучения нейронной сети необходимо сформировать обучающую выборку. Для этого необходимо генерировать тренировочные наборы, которые отражают наиболее типичные участки траектории. Выходной вектор должен соответствовать при этом желаемому углу поворота робота (рис.4.8).

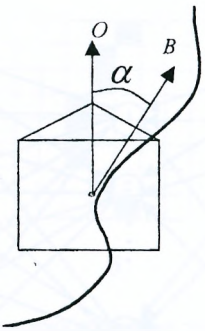


Рис. 4.8. Определение желаемого направления робота: О – текущее направление робота; В – желаемое направление; $\alpha=20^o$ – требуемый угол поворота

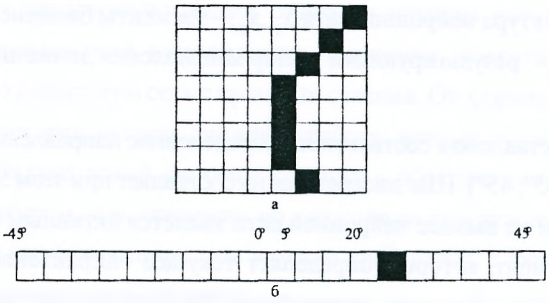


Рис. 4.9. Формирование обучающего паттерна: а – входной паттерн; б – выходной паттерн

Для представленного на рис 4.8 фрагмента траектории, обучающий паттерн приведен на рис.4.9.

С целью обучения нейронной сети были разработаны обучающие наборы, которые можно условно разделить на два класса. Первый класс представляет собой фрагменты прямых линий, наблюдаемые под различными

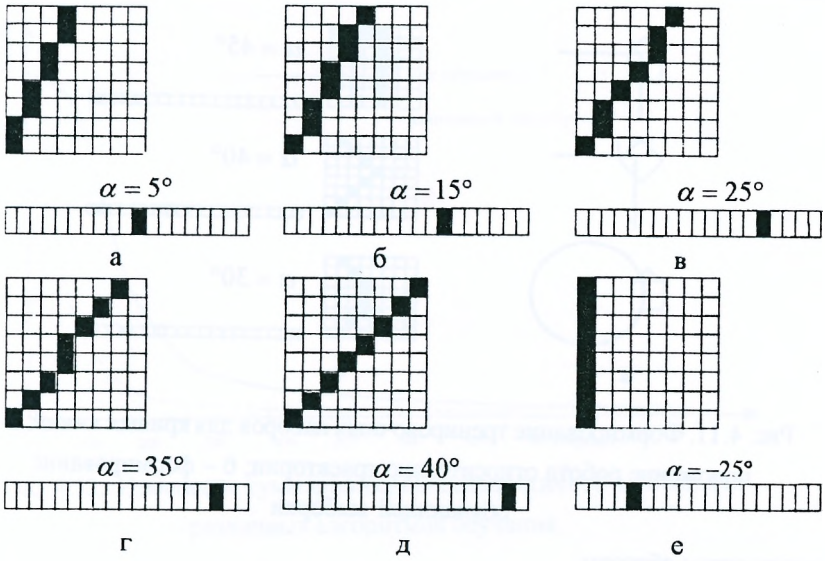


Рис. 4.10. Пример формирования шести тренировочных наборов а, б, в, г, д, е для фрагментов прямой линии

углами и рекомендуемые углы поворота для каждого фрагмента. Примеры тренировочных наборов для данного класса приведены на рис.4.10.

Второй класс тренировочных наборов представляет собой фрагменты кривых линий, которые имеют различную степень кривизны. Слева на рис.4.11 изображено положение робота относительно заданной траектории, а справа - обучающие паттерны.

Всего было получено 64 тренировочных набора. Для обучения нейронной сети применяется алгоритм обратного распространения ошибки. При этом проводились эксперименты с использованием постоянного и адаптивного шага обучения, где адаптивный шаг обучения определяется

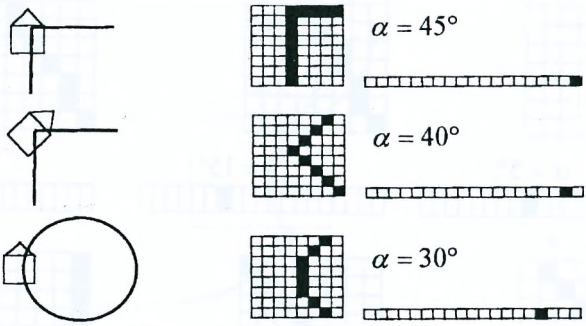


Рис. 4.11. Формирование тренировочных наборов для кривых линий; а – положение робота относительно траектории; б – формирование обучающей выборки

следующим образом:

$$\alpha(t) = \frac{4\sum_j \gamma_j^2 y_j (1 - y_j)}{(1 + \sum_i y_i^2)\sum_j \gamma_j^2 y_j^2 (1 - y_j)^2}, \quad (4.1)$$

где i, j характеризуют слои нейронных элементов, между которыми производится модификация синаптических связей; γ_j - ошибка соответствующего нейронного элемента.

Графики изменения суммарной среднеквадратичной ошибки для различных алгоритмов приведены на рис. 4.12.

Из рисунка следует, что адаптивный шаг обучения позволяет уменьшить суммарную среднеквадратичную ошибку сети и увеличивает быстродействие

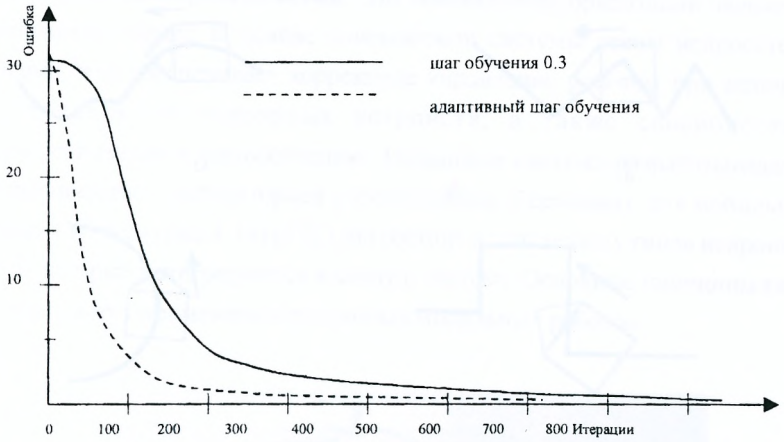


Рис. 4.12. Изменение суммарной среднеквадратичной ошибки для различных алгоритмов обучения

процедуры обучения. Так, использование адаптивного шага позволило достичь ошибки $8*10^{-4}$ за 1200 итераций, а при использовании a=0.3 за 4755 итераций достигалась ошибка $12*10^{-4}$.

4.1.4. Тестирование

С целью проведения экспериментов разработано программное обеспечение для эмуляции нейронной сети и моделирования движения робота. При этом на экране компьютера может задаваться произвольная траектория движения робота. Компьютерное моделирование показало устойчивое движение робота по произвольным траекториям с различными углами поворота (рис.4.13).

Это происходит за счет обобщения нейронной сетью знаний на неизвестные типы трасс, которые не входили в обучающую выборку.

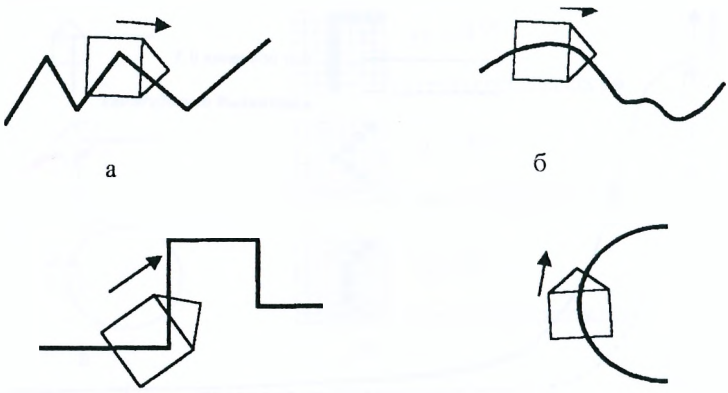


Рис. 4.13. Примеры движения робота по различным траекториям: а– движение робота по ломаной кривой; б– движение робота по извилистой кривой ; в– движение робота по прямоугольной линии; г– движение робота по окружности

Моделирование показало устойчивое функционирование такой системы. Так при любых отклонениях робота от заданной траектории в пределах видимости видеокамеры, он возвращается на требуемый путь.

4.2. Автономное управление мобильным роботом

В данном разделе описываются основные концепции и функционирование нейронной системы для автономного управления мобильным роботом [30-39]. При этом предполагается, что движение робота осуществляется в неизвестном пространстве. Задача робота состоит в том, чтобы зная координаты целевой точки, достичь конечного пункта движения

в пространстве с препятствиями. Это эквивалентно ориентации человека в незнакомом городе. В основе описываемой системы лежит нейросетевой аппарат, что обеспечивает корректное управление роботом при неточной информации от сенсорных устройств, а также способность к самоорганизации и самообучению. Нейронная система разрабатывалась в сотрудничестве с лабораторией робототехники (Германия) для мобильного робота «Walter» (рис.4.14) [27]. Она состоит из различных типов нейронных сетей, которые интегрируются в единую систему. Основные принципы такой системы могут применяться для любых мобильных роботов.

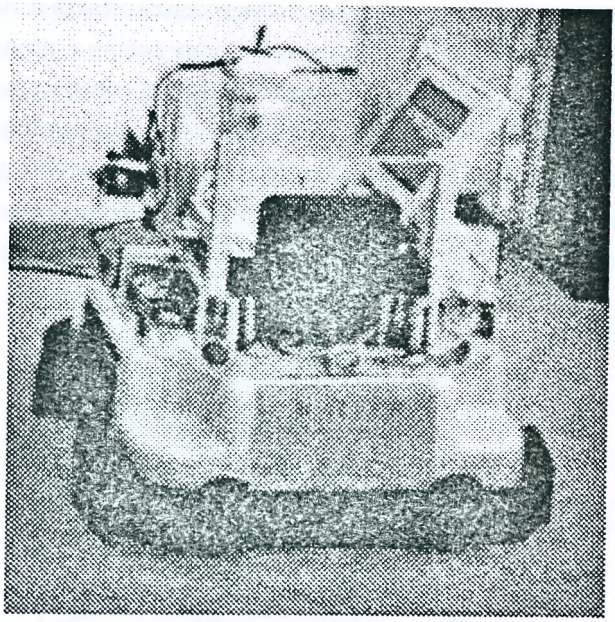


Рис. 4.14. Мобильный робот "Walter"

4.2.1. Входная информация и задачи нейронной системы

Для отображения окружающей обстановки мобильные роботы оборудованы различными сенсорными устройствами. В качестве сенсорных устройств на роботе «Walter» используются как недорогие ультразвуковые датчики и 2D инфракрасный сканер, так и видеокамера. Ультразвуковые датчики установлены по периметру робота согласно рис.4.15.

Каждый из них характеризуется частотой 35 кГц и углом излучения 20°. Инфракрасный сканер расположен по фронту робота. В качестве него используется сканер RS2-180 фирмы «Leuze electronic». Робот «Walter» имеет квадратную форму, на бамперах которого расположены тактильные датчики. Они предназначены для фиксации столкновения робота с препятствиями. Кроме этого робот оборудован портативным компьютером, а также телекоммуникационной аппаратурой для связи со станцией «Sun». Управление роботом может производиться как в операторном режиме при помощи джойстика, так и автономно при помощи соответствующей системы. Это создает хорошую базу для проведения экспериментальных исследований.

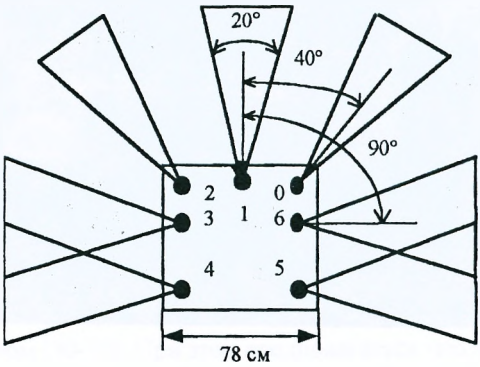


Рис.4.15. Схема расположения ультразвуковых датчиков

Описываемая нейронная система использует в качестве входных данных только информацию от ультразвуковых датчиков и инфракрасного сканера. Задача ее состоит в том, чтобы на основе информации от разнородных сенсорных устройств и координат конечной точки формировать оптимальное направление движения в пространстве с препятствиями. Это эквивалентно обеспечению минимального угла а между направлением на цель и текущим направлением робота (рис.4.16).

Для определения положения робота относительно цели используется подвижная и неподвижная система координат. Центр неподвижной системы совпадает с начальной точкой движения робота. Подвижная система координат проходит через центр робота и ось ординат ее совпадает с текущим направлением робота (рис.4.16). В дальнейшем подвижную систему будем называть системой координат робота.

В общем случае нейронная система решает следующие задачи:

–генерация на каждом шаге направления и скорости движения робота

–обеспечение робастного управления роботом при неточной информации от сенсорных устройств

–обучение с учителем

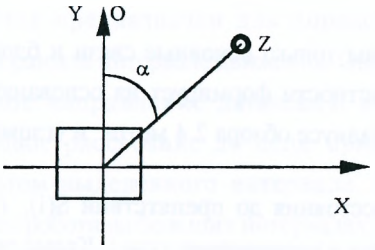


Рис.4.16. Положение робота относительно цели:
O - текущее направление; Z – целевая точка

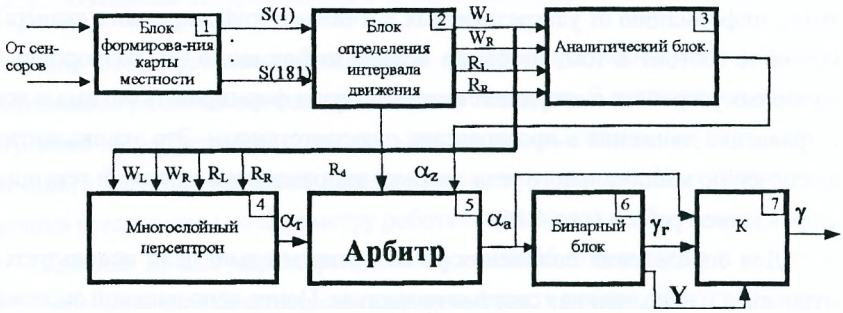


Рис 4.17. Архитектура нейронной системы

– самообучение с целью самоорганизации в процессе функционирования

– функционирование в реальном масштабе времени.

4.2.2. Архитектура нейронной системы

Общая архитектура нейронной системы для автономного управления движением робота изображена на рис.4.16. Она состоит из различных типов нейронных сетей.

На рисунке показаны только основные связи и блоки системы. Блок формирования карты местности формирует на основании информации от сенсорных устройств в радиусе обзора 2.4 метра и угловом диапазоне 180° линейные и угловые расстояния до препятствий S(i), $i = \overline{1,181}$, где S - расстояние до препятствия в направлении угла i. Кроме этого он генерирует сжатую карту местности S(p), $p = \overline{1,36}$, которая используется для управления бинарным блоком.

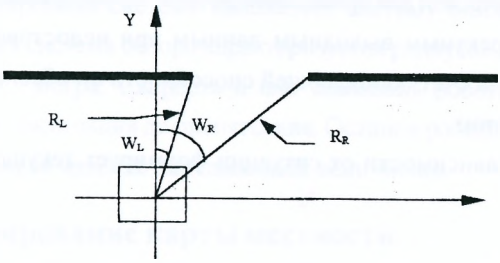


Рис.4.18. Расположение интервала движения относительно робота

Блок формирования интервала движения предназначен для выделения в окружающем пространстве с препятствиями оптимального промежутка движения. Оптимальный промежуток движения характеризуется тем, что он находится наиболее близко к цели. Информация на выходе данного блока соответствует линейным (R_L, R_R) и угловым (W_L, W_R) расстояниям выделенного интервала движения (рис.4.18).

В случае, когда не выделен свободный интервал движения, то происходит поворот робота на 90°, если это возможно, и поиск свободного интервала, или осуществляется движение робота назад для выхода из тупика. Аналитический блок предназначен для определения оптимального направления α_z в выбранном интервале движения. Оптимальное направление характеризует такое направление движения, которое обеспечивает минимальное угловое расстояние до цели при условии свободного прохождения роботом выделенного интервала. Аналитический блок управляет движением робота на больших интервалах движения, когда R_d>2d. Здесь R_d - ширина выделенного интервала движения, а d - ширина робота.

Многослойный персептрон предназначен для ориентации робота на узких интервалах движения, где R_d<2d. Он формирует робастное направление движения робота α_r. На узких интервалах движения неточная карта местности

становится критической для ориентации робота. Если обучить многослойный персептрон корректным выходным данным при недостоверной входной информации, то за счет обобщающей способности он обеспечит робастное управление роботом.

Арбитр в зависимости от ситуации формирует текущее направление робота:

$$\alpha_a = \begin{cases} \alpha_z, если Rd > 2d \\ \alpha_r, иначе \end{cases} \quad (4.2)$$

Бинарный блок служит для управления роботом в ситуации, когда боковое расстояние до препятствия Δ является слишком маленьким для осуществления резких поворотов. Для робота «Walter» Δ=17см. Данный блок преобразует входную информацию в бинарный массив. Направление, которое формирует бинарный блок, не превышает 1°. Это обеспечивает исключение контакта робота с боковыми препятствиями. Коммутатор в зависимости от ситуации формирует окончательное направление движения робота:

$$\gamma = \begin{cases} \alpha_a, если\ Y = 0 \\ \gamma_r, иначе \end{cases} \quad (4.3)$$

где Y=1, если $\Delta \leq$17см.

Таким образом в зависимости от ситуации роботом могут управлять следующие модули:

- аналитический блок
- многослойный персептрон
- бинарный блок совместно с аналитическим
- бинарный блок совместно с многослойным персептроном.

Такой подход обеспечивает устойчивое движение робота в различных

ситуациях. Нейронная система использует систему ближнего и дальнего обзора. Дальняя система обзора характеризуется радиусом обзора 2.4 метра, а ближняя - 0.7 метра. Скорость и шаг движения робота нормируются в зависимости от расстояния до препятствия. Останов робота происходит, если расстояние до цели меньше определенной величины ε.

4.2.3. Формирование карты местности

Для ориентации робота в пространстве необходимо обрабатывать информацию от разнородных сенсорных устройств с целью получения единой интеграционной картины окружающей обстановки. Такая картина называется картой местности (occupancy grid). Она характеризует в определенном радиусе обзора и угловом диапазоне 180° расположение препятствий и расстояния до них:

$$OG = \{S(i), -90° \le i \le 90°\}, \qquad (4.4)$$

где S(i) – расстояние до препятствия, если угол между текущим направлением робота и препятствием равен i градусов.

Шаг изменения угла i равен 1°. Существуют различные подходы к формированию карты местности. Так как система управления роботом должна работать в реальном масштабе времени, то основным критерием здесь является минимальная сложность алгоритма, даже в ущерб точности. Тем более, как будет показано далее, нейросетевой подход позволяет обеспечить устойчивое управление роботом, даже при неточной карте местности.

Рассмотрим простой аналитический подход к формированию карты местности. Входной информацией здесь являются данные от ультразвуковых датчиков и 2D инфракрасного сканера. На роботе "Walter" установлено 7 ультразвуковых датчиков (рис.4.15), каждый из которых характеризуется

частотой 35кГц и углом излучения 20°. Это не позволяет точно определить угловое расстояние до препятствия.

В качестве инфракрасного сканера на данном роботе применяется сканер RS2-180 фирмы "Leuze electronic". Область обзора его характеризуется радиусом от 0.3 до 2 метров. Инфракрасный сканер разделяет пространство обзора равное 180° на 32 сектора. Угловой диапазон одного сектора составляет приблизительно 5.6°. Точность информации от сканера зависит от различных внешних факторов и от типа отражающей поверхности. В любом случае погрешность определения сканером линейных расстояний до препятствия значительно выше по сравнению с ультразвуковыми датчиками, которые характеризуются устойчивой работой в диапазоне до 10 метров. Преимуществом сканера является то, что он формирует рельеф окружающей обстановки. Это позволяет с определенной степенью достоверности формировать угловые характеристики препятствий. Алгоритм формирования карты местности основан на том, что ультразвуковые датчики формируют достоверные линейные, а инфракрасный сканер позволяет определить угловые расстояния до препятствий.

Рассмотрим общие принципы формирования карты местности. Она характеризуется радиусом 2.4 метра от центра робота и угловым диапазоном 180°.

Инфракрасный сканер производит обзор пространства слева направо. Соответствующим образом нумеруются сектора сканера, которые будем называть элементами. Так первый элемент сканера характеризует угловой диапазон от 0 до 5.6 градусов. Карту местности робота необходимо формировать относительно центра его координат. При отображении секторов сканера в центр робота получается в угловом диапазоне 180° количество секторов равное 36, каждый из которых имеет размер 5°. При этом элементы сканера соответствуют диапазону от 3-го до 34-го сектора. Пусть l – номер

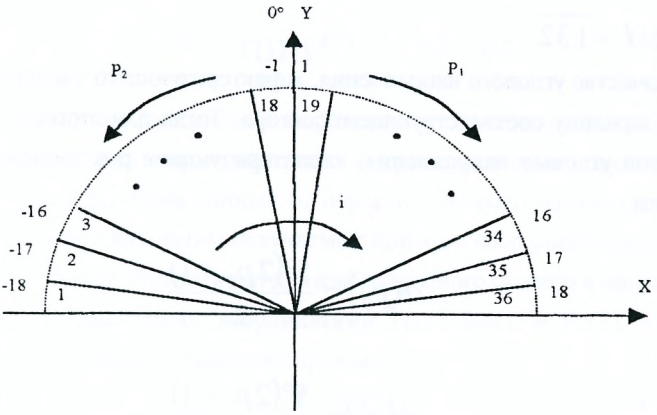


Рис.4.19 Нумерация секторов сканера

сектора в системе координат сканера. Тогда номер сектора сканера относительно центра робота равен:

$$i = l + 2, \tag{4.5}$$

где $l = \overline{1,32}$.

В системе координат робота будем использовать нумерацию секторов относительно его текущего направления (рис.4.18). Тогда в первом квадранте секторы нумеруются от 1 до 18, а во втором квадранте от –1 до –18. Для перевода секторов сканера в секторы работа соответственно для второго и первого квадранта используются следующие выражения:

$$p_2 = (i - 19) \bmod 19, \tag{4.6}$$

$$p_1 = (i + 1) \bmod 19, \tag{4.7}$$

где $i=l+2$, $l=\overline{1,32}$.

В качестве углового направления, характеризующего элемент сканера, примем середину соответствующего сектора. Тогда для второго и первого квадрантов угловые направления, характеризующие p-й элемент сектора равняются:

$$\alpha(p_2)=\frac{5^\circ(2p_2+1)}{2} \tag{4.8}$$

$$\alpha(p_1)=\frac{5^\circ(2p_1-1)}{2} \tag{4.9}$$

Приведенные выше формулы позволяют однозначно перевести номер сектора сканера в номер сектора и угловое направление робота. Пусть U(j) - характеризует линейное расстояние до препятствия, полученное от j-го датчика, а C(j, p) - p-й элемент сканера, который принадлежит зоне действия j-го датчика и соответствует линейному расстоянию до препятствия в угловом направлении a(p).

Рассмотрим формирование карты местности между вторым и первым (рис.9.15) ультразвуковыми датчиками. Область действия второго датчика может охватывать элементы сканера от -6 до -9, а первого датчика - от -2 до 2. В результате получается информация U(2), U(1) от датчиков и C(2, p), $p=-\overline{6,-9}$; C(1, p), $p=-\overline{2,2}$ от сканера. Для каждого датчика определяем номер элемента сканера, который соответствует минимальному расстоянию

$$k_1=\min_p|U(1)-C(1,p)|, \tag{4.10}$$

$$k_2 = \min_p |U(2) - C(2,p)| \qquad (4.11)$$

Элементы k_1 и k_2 однозначно определяют угловые направления $\alpha(k_1)$ и $\alpha(k_2)$, которые характеризуют угловые расстояния до препятствий. Производим отображение данных от первого и второго датчика в систему координат робота. Обозначим получаемые при этом координаты через Yu(1), Xu(1) и Yu(2), Xu(2). Аналогично для инфракрасного сканера в направлении $\alpha(k_1)$ и $\alpha(k_2)$ имеем точки с координатами Yc(1), Xc(1) и Yc(2), Xc(2). По двум точкам составляем уравнения прямых:

$$Yu = f(Xu) \qquad (4.12)$$

$$Yc = f(Xc) \qquad (4.13)$$

Для каждого элемента сканера p, где $p = \overline{k_1, k_2}$, определяем координаты точек пересечения прямой ОС, которая имеет наклон $\alpha(p)$, с прямыми Yu=f(Xu) и Yc=f(Xc) (рис.4.20).

В результате для каждого углового направления $\alpha(p)$, где $p = \overline{k_1, k_2}$, получаются точки с координатами Xu(p), Yu(p) и Xc(p), Yc(p). Вычисляем расстояние между ними:

$$\delta(p) = b\sqrt{(X_u(p) - Xc(p))^2 + (Y_u(p) - Yc(p))^2}, \quad (9.14)$$

$$b = \begin{cases} 1, если \sqrt{X_c^2(p) + Y_c^2(p)} < \sqrt{X_u^2(p) + Y_u^2(p)} \\ -1, иначе \end{cases}, \quad (9.15)$$

где $p = \overline{k_1, k_2}$.

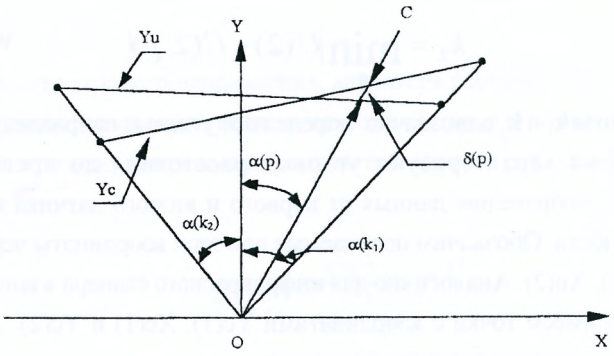


Рис.4.20 Определение разности δ(p) между показаниями датчиков и сканера

Путем коррекции данных от сканера производится построение карты местности в соответствующей области. При этом если показания от ультразвуковых датчиков и сканера больше 2.4 метра, то корректировка данных от сканера не осуществляется. В противном случае определяется p-й элемент карты местности следующим образом:

$$S(p)=C(p)+\delta(p), \qquad (4.16)$$

где $p = \overline{k_1, k_2}$; C(p) – линейное расстояние до препятствия в направлении p, которое получается от инфракрасного сканера.

В результате получается карта местности в диапазоне между 1 и 2 датчиком. Аналогичный принцип применяется при построении карты местности в области между нулевым и первым датчиками.

В диапазоне между 2 и 3 датчиками информация от сканера не отображается в область действия третьего датчика. Поэтому здесь применяется несколько иной подход. В соответствии с ним определяется расстояние $\delta(k_2)$ только в точке k_2, которая характеризует элемент сканера

наиболее близкий по расстоянию к показаниям второго датчика (рис. 4.15). Если показания второго датчика и сканера более 2.4 метра, то коррекция не происходит. В противном случае для каждого элемента сканера в соответствующей области выполняются следующие действия:

$$S(p)=C(p)+\delta(\kappa_2), \qquad (4.17)$$

где $p = \overline{\kappa_2,-16}$.

Аналогичным образом формируется карта местности в области между

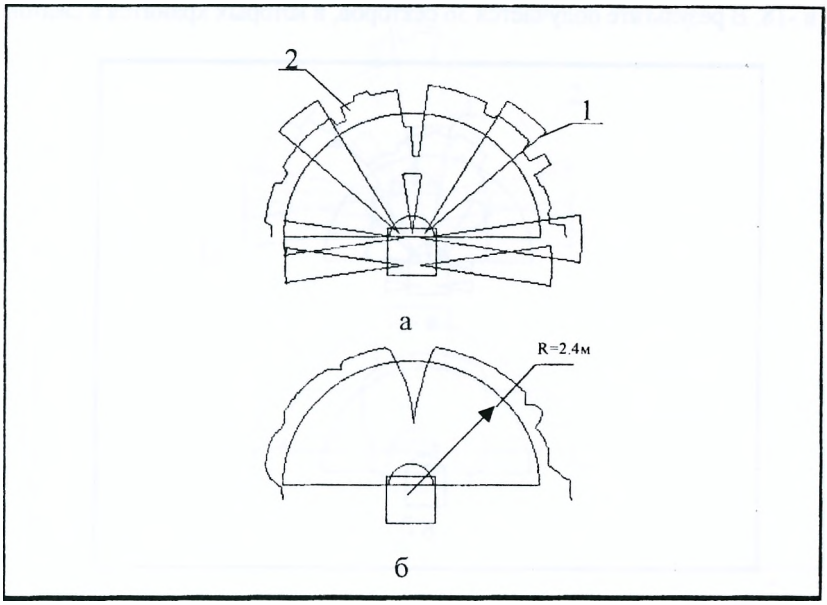


Рис. 4.21. Формирование карты местности , когда по фронту робота находится столб прямоугольной формы: а – входная информация от сенсорных устройств; б – карта местности; 1–информация от ультразвуковых датчиков; 2–информация от инфракрасного сканера

нулевым и шестым датчиком. В результате этого определяется диапазон секторов карты местности от -16 до 16, что соответствует угловым направлениям от -77.5 до 77.5 градусов.

Определим оставшиеся элементы карты местности. Сектор 17 заполняется информацией, характеризующей расстояние до препятствия полученное от шестого датчика (рис. 4.15). В качестве данных для 18-го сектора используется среднее значение расстояний, полученных от пятого и шестого датчиков. Аналогичным образом происходит заполнение секторов с номерами -17 и -18. В результате получается 36 секторов, в которых хранится в сжатой

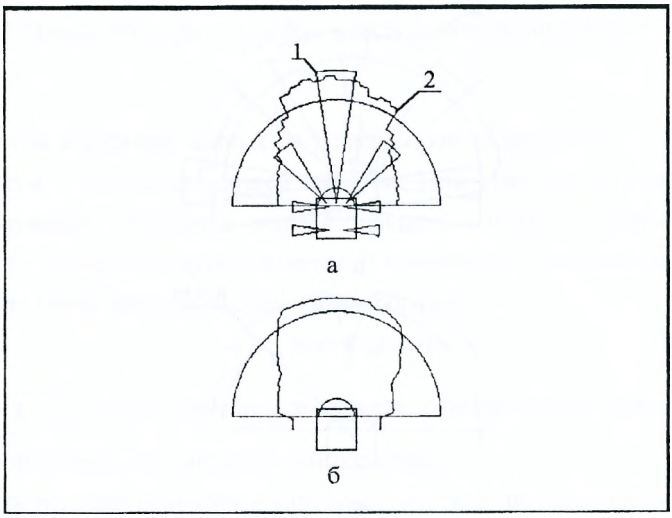


Рис. 4.22. Формирование карты местности при движении робота по коридору; а – входная информация от сенсорных устройств; б – карта местности; 1–информация от ультразвуковых датчиков; 2–информация от инфракрасного сканера

форме информация, характеризующая расстояния до препятствий. Для получения карты местности состоящей из 181 элемента необходимо провести дополнительные преобразования. Для этого все угловые направления двух крайних секторов (18 и -18) с шагом 1° заполняются линейными расстояниями, соответствующими этим секторам. Для остальных секторов линейные расстояния соответствуют их серединам. В результате получается совокупность

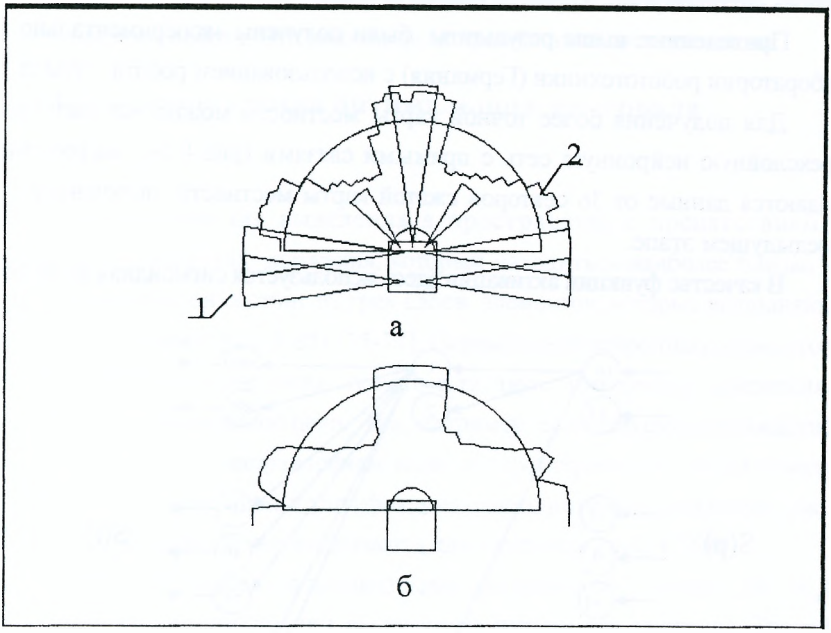


Рис. 4.23. Формирование карты местности, когда по фронту робота находится проем двери: а – входная информация от сенсорных устройств; б – карта местности; 1–информация от ультразвуковых датчиков; 2–информация от инфракрасного сканера

точек S(p) и α(p), где $p = \overline{-17,17}$. Соединяя полученные точки отрезками прямых и ограничивая область обзора радиусом 2.4 метра получаем карту местности. На основе рассмотренного подхода разработано программное обеспечение для формирования карты местности, которое апробировано на мобильном роботе «Walter». Примеры формирования карты местности для разных ситуаций изображены на рис.4.21 – 4.23.

Приведенные выше результаты были получены экспериментально в лаборатории робототехники (Германия) с использованием робота "Walter".

Для получения более точной карты местности можно использовать трехслойную нейронную сеть с прямыми связями (рис.4.24), на которую подаются данные от 36 секторов сжатой карты местности, полученной на предыдущем этапе.

В качестве функции активации здесь используется сигмоидная функция.

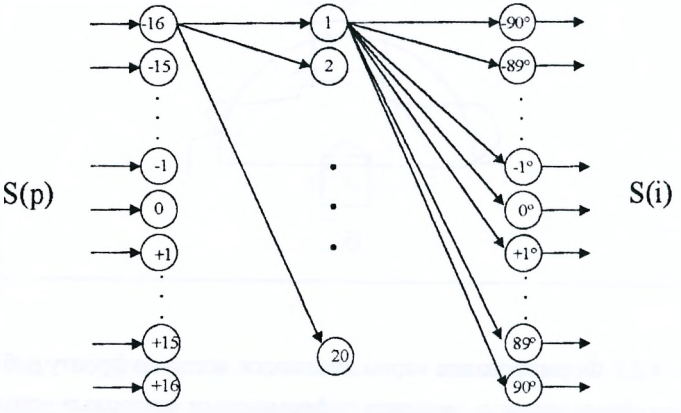


Рис.4.24 Трехслойная нейронная сеть

Поэтому входные данные необходимо отобразить в отрезок [0, 1]. Для формирования обучающей выборки используется вращение полученных экспериментально эталонных данных в диапазоне 180°.

Рассмотренный подход к формированию карты местности характеризуется простотой и возможностью работы робота в реальном масштабе времени. Точность карты местности является достаточной для того, чтобы робот проходил узкие участки, ширина которых, например, на 10 см больше ширины робота. Это подтвердили проведенные эксперименты.

4.2.4 Блок определения оптимального интервала движения

Предназначен для выделения в пространстве с препятствиями свободного промежутка движения, который находиться наиболее близко к цели. Структурно он состоит из трех слоев элементов, которые выполняют различные функции (рис. 4.25) [35-37]. Первый слой нейронных элементов предназначен для выделения одного или двух интервалов движения, ближайших к направлению цели. Так, например, если текущее направление робота совпадает с направлением цели и характеризуется отсутствием препятствия, то происходит идентификация одного интервала движения (рис. 4.25 а). В противном случае выделяются два интервала (рис. 4.25 б).

Второй слой элементов предназначен для фиксации угловых (W_L, W_R) и линейных (R_L, R_R) расстояний выделенных интервалов движения. Третий слой состоит из одного процессорного элемента, который производит анализ возможности прохождения роботом идентифицированных интервалов движения. Если существуют два интервала движения, то процессорный элемент определяет среди них наиболее близкий к цели.

Рассмотрим подробно функционирование различных слоев блока

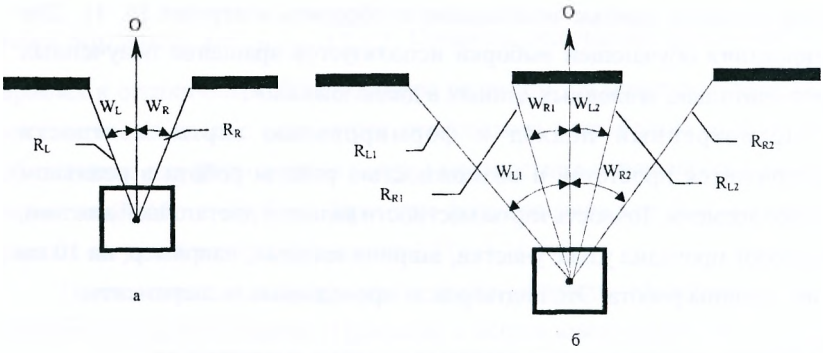


Рис. 4.25 Идентификация интервалов движения робота: а – один интервал движения; б – два интервала движения

определения интервала движения. Первый слой содержит 181 пороговых нейронных элемента, которые соединены между собой горизонтальными связями (рис.4.26).

Каждому нейронному элементу поставлен в соответствие определенный элемент S(i), где $-90° \le i \le 90°$, карты местности. Так текущему направлению робота соответствует нулевой нейронный элемент, который топологически расположен в середине первого слоя. Нейроны которые расположены по краям, характеризуются направлением -90° и 90° (рис. 4.27).

Входной информацией первого слоя является карта местности, каждый элемент S(i) которой поступает на соответствующий нейронный элемент. В качестве пороговых нейронных элементов используются три типа нейронов: направляющие, промежуточные и концевые. Они выполняют функции логической обработки информации от карты местности. Направляющий нейронный элемент (рис. 4.28.) предназначен для возбуждения остальных нейронов первого слоя.

Он формирует следующие сигналы:

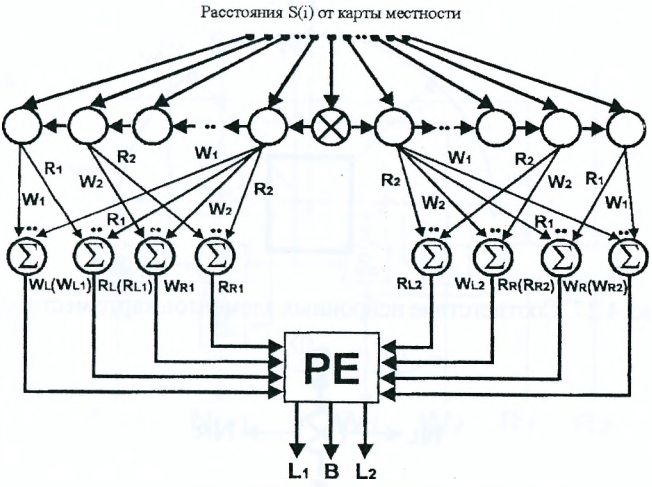


Рис. 4.26. Архитектура блока определения оптимального интервала движения

$$N_L(N_R) = \begin{cases} 1, & \textit{если } S \geq R_t \\ 0, & \textit{иначе} \end{cases} \qquad (4.18)$$

$$F_L(F_R) = \begin{cases} 1, & \textit{если } S < R_t \\ 0, & \textit{иначе}, \end{cases} \qquad (4.19)$$

где S – расстояние до препятствия, поступающее на направляющий нейрон, R_t предельный порог видимости, который определяется системой обзора робота. Для дальней системы обзора R_t = 2.4м, а для ближней – R_t = 0.8м от центра робота.

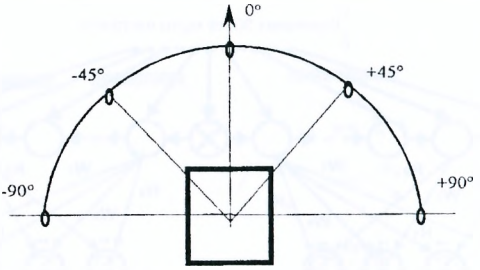


Рис. 4.27. Соответствие нейронных элементов карте местности

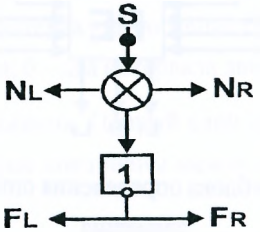


Рис. 4.28. Направляющий нейронный элемент

Единичные значения N_L и N_R характеризуют, что в соответствующем направлении в пределах видимости системы обзора не существует препятствия. При этом происходит выделение одного интервала движения. Единичные значения F_L и F_R возбуждают остальные нейроны первого слоя для поиска свободных интервалов движения слева и справа.

Промежуточный нейронный элемент (рис. 4.29) предназначен для определения угловых (W_1, W_2) и линейных (R_1, R_2) расстояний, которые характеризуют выделенный интервал движения.

Кроме этого он формирует сигналы возбуждения (F_{i+1}, N_{i+1}) следующего нейронного элемента. На входы промежуточного нейрона с номером i+1 поступают сигналы возбуждения F_i и N_i, а также расстояние до препятствия S(i) от предыдущего элемента. Помимо этого входом его является расстояние

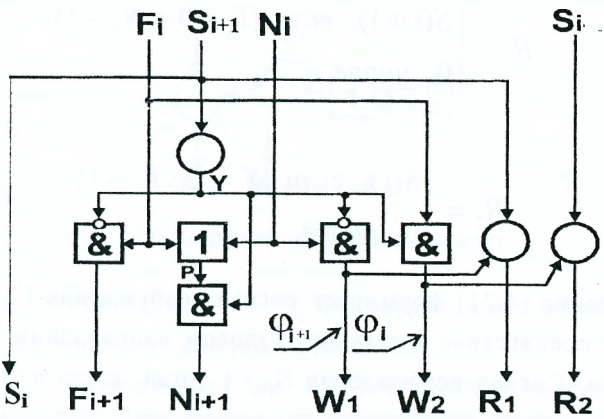


Рис. 4.29.Промежуточный нейронный элемент

S(i+1), поступающее от карты местности. Промежуточный нейронный элемент выполняет следующие функции:

$$Y = \begin{cases} 1, & \textit{если } S \geq R_t \\ 0, & \textit{иначе}, \end{cases} \tag{4.20}$$

$$F_{i+1} = \begin{cases} 1, & \textit{если } (Y = 0 \wedge Fi = 1) \\ 0, & \textit{иначе}, \end{cases} \tag{4.21}$$

$$N_{i+1} = \begin{cases} 1, & \textit{если } (Y = 1 \wedge P = 1) \\ 0, & \textit{иначе}. \end{cases} \tag{4.22}$$

$$R_1 = \begin{cases} S(i+1), \ \textit{если} \ (Y = 0 \wedge N_i = 1) \\ 0, \ \textit{иначе}. \end{cases} \quad (4.23)$$

$$R_2 = \begin{cases} S(i), \ \textit{если} \ (Y = 1 \wedge F_i = 1) \\ 0, \ \textit{иначе}. \end{cases} \quad (4.24)$$

Выражение (4.21) формирует сигнал возбуждения $F_{i+1}=1$, если существует препятствие в текущем угловом направлении (i+1) и в предыдущем. Сигнал возбуждения $N_{i+1}=1$ тогда, когда в текущем и предыдущем угловом направлении нет препятствий. Выражение (4.23) формирует линейное расстояние до препятствия $R_1=S(i+1)$, если в предыдущем направлении i не было, а в текущем направлении i+1 имеется препятствие. Сигнал $R_2=S(i)$ формируется тогда, когда предыдущее направление характеризуется наличием препятствия, а текущее – его отсутствием.

Угловые направления W_1 и W_2 формируются следующим образом:

$$W_1 = \begin{cases} \varphi_{i+1}, \ \textit{если} \ (Y = 0 \wedge N_i = 1) \\ 0, \ \textit{иначе}, \end{cases} \quad (4.25)$$

$$W_2 = \begin{cases} \varphi_i, \ \textit{если} \ (Y = 1 \wedge F_i = 1) \\ 0, \ \textit{иначе}, \end{cases} \quad (4.26)$$

где φ_{i+1}, φ_i – соответственно весовые коэффициенты промежуточного нейронного элемента, которые равняются угловому направлению i+1 и i.

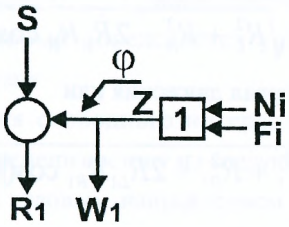


Рис. 4.30.Концевой нейронный элемент

Заметим, что сигналы W_2 и R_2 формируются в том случае, если происходит выделение двух интервалов движения.

Концевой нейронный элемент (рис.4.30) предназначен для формирования сигналов R_1 и W_1, если на предыдущих этапах обработки информации не был выделен интервал движения.

Он выполняет следующие функции:

$$R_1 = \begin{cases} \min\{S, R_t\}, если.z = 1 \\ 0, иначе \end{cases}, \qquad (4.27)$$

$$W_1 = z\varphi, \qquad (4.28)$$

где $\varphi = \pm 90°$ (рис.4.27).

Второй слой состоит из суммирующих элементов (рис.4.26), которые выделяют угловые и линейные расстояния до препятствий. При этом каждый нейронный элемент первого слоя (за исключением направляющего нейрона) имеет связь со всеми элементами второго слоя. Процессорный элемент предназначен для определения возможности прохождения роботом выделенных интервалов движения и выбора оптимального интервала. Для этого он определяет ширину расстояния между препятствиями.

$$R_d = \sqrt{R_L^2 + R_R^2 - 2R_L R_R \cos(|W_R| + |W_L|)}, \quad (4.29)$$

если выделен один интервал движения или

$$R_{d1} = \sqrt{R_{L1}^2 + R_{R1}^2 - 2R_{L1}R_{R1} \cos(|W_{R1}| + |W_{L1}|)}, \quad (4.30)$$

$$R_{d2} = \sqrt{R_{L2}^2 + R_{R2}^2 - 2R_{L2}R_{R2} \cos(|W_{R2}| + |W_{L2}|)}, \quad (4.31)$$

если существует два возможных интервала движения робота.

Если выделен один интервал движения, то процессорный элемент сравнивает ширину интервала R_d с шириной робота d и формирует сигнал B:

$$B = \begin{cases} 1, если (R_d > d) \\ 0, иначе \end{cases} \quad (4.32)$$

Если выделены два интервала движения робота, то процессорный элемент кроме этого выбирает среди них оптимальный, который соответствует наименьшему угловому расстоянию до цели. Для этого он формирует сигналы L_1 и L_2 возбуждения соответствующего интервала движения:

$$L_1 = \begin{cases} 1, если (R_{d1} > d) \,\&\, (W_{R1} < W_{L2}) \\ 0, иначе \end{cases} \quad (4.33)$$

$$L_2 = \begin{cases} 1, если (R_{d2} > d) \,\&\, (W_{R1} < W_{L2}) \\ 0, иначе \end{cases} \quad (4.34)$$

Таким образом сигналы B, L_1, L_2 идентифицируют соответствующий интервал движения, который определяется угловыми и линейными расстояниями до препятствия.

Алгоритм выделения свободного интервала движения робота в фиксированный момент времени состоит из следующих шагов:

1. В соответствии с угловым направлением до цели a выбирается направляющий нейронный элемент с номером k, принадлежащий текущей карте местности и $-90°<k<90°$, так чтобы обеспечивалось выполнение следующего условия:

$$|\alpha-k| \to \min$$

2. В результате взаимодействия нейронных элементов между собой происходит выделение потенциальных интервалов движения. Так, если $S(k) \geq R_1$, то выделяется один интервал движения. В противном случае могут выделиться два возможных интервала движения $[W_{L1}, W_{R1}]$, $[W_{L2}, W_{R2}]$ слева и справа от направляющего нейронного элемента.

3. Процессорный элемент производит анализ возможности прохождения роботом выбранных интервалов движения и в соответствии с угловым расстоянием до цели выбирает оптимальный интервал.

4. Если в результате выполнения пункта 3 не существует подходящего интервала движения, то выбирается следующий направляющий элемент слева и справа нейронной цепочки:

$$K_L = W_{L1} - 1$$
$$K_R = W_{R2} + 1$$

где K_L и K_R - соответственно номера направляющих нейронных элементов, которые расположены слева и справа нейронной цепочки. При этом распространение информации от направляющих элементов происходит только в одну сторону.

5. Процесс повторяется начиная с пункта 2 до тех пор, пока не произойдет

выделение интервала движения робота.

В случае, если интервал движения не выделен или $R_F < S(\alpha)$, где R_F - расстояние от робота до цели, $S(\alpha)$ - расстояние до препятствия в направлении цели, то включается ближняя система обзора. Данная система обзора включается также, когда линейные характеристики интервала движения (R_L, R_R) робота меньше определенной величины. Следует также отметить, что в начальный момент движения робота он ориентируется в направлении цели. Приведенный выше алгоритм позволяет не анализировать все пространство решений, определяемое картой местности. Он выбирает всегда ближайший к цели свободный интервал движения. Это напоминает процесс ориентации человека при поиске цели в незнакомом пространстве.

Рассмотренная сеть является динамической нейронной сетью с фиксированными весовыми коэффициентами, которые характеризуются угловым положением нейронных элементов. Динамичность ее заключается в варьировании направляющих нейронных элементов в зависимости от ситуации.

4.2.5. Аналитический блок

Предназначен для определения в выделенном интервале движения робота оптимального направления. Для этого он сканирует оптимальный интервал движения и в соответствии с шириной робота, а также кратчайшим расстоянием до цели ($\min \alpha$) формирует оптимальное направление движения робота. Архитектура данного блока состоит из различных слоев нейронных и процессорных элементов (рис 4.31), которые выполняют различные функции.

Входной информацией аналитического блока является угол а между текущим направлением и целью, а также угловые (W_L, W_R) и линейные (R_L, R_R) характеристики выделенного интервала движения. Первый слой состоит из двух процессорных элементов (PE_1, PE_2), которые предназначены для

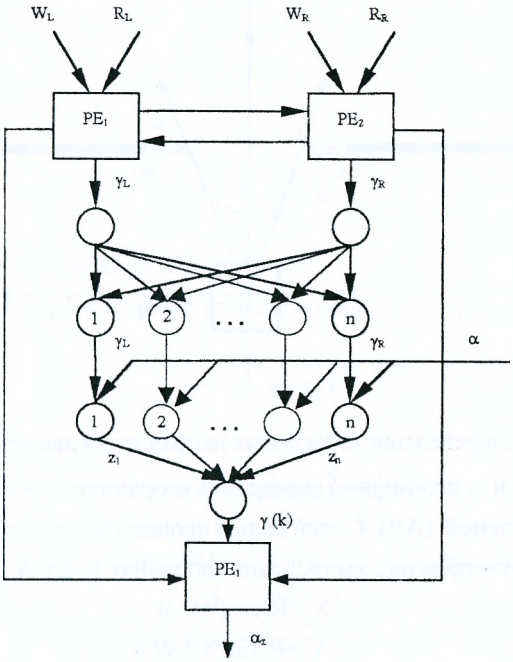


Рис. 4.31 Архитектура аналитического блока

определения в интервале [A, B] таких направлений движения γ_L и γ_R, при которых не происходит столкновение с левой и правой кромкой препятствия (рис. 4.32).

Для этого необходимо, чтобы выполнялось следующее условие:

$$S_R(S_L)>d/2 \qquad (4.35)$$

где d - ширина робота.

Такое условие соответствует делению отрезка [A, B] в заданном соотношении. Введем декартову систему координат, ось ординат которой совпадает с текущим направлением робота O. (рис. 4.32). Для вычисления

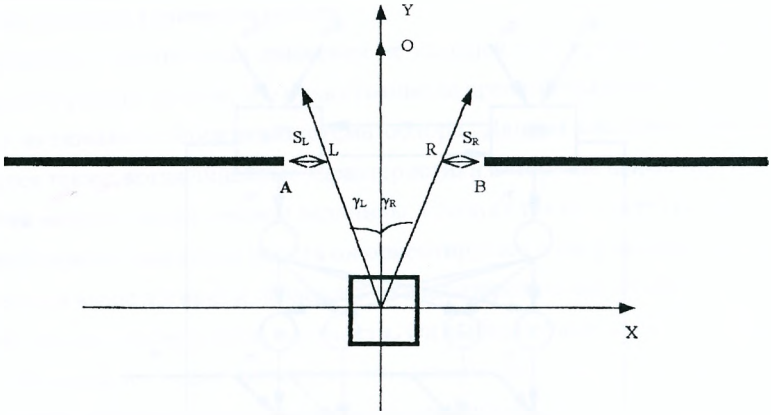


Рис. 4.32. Определение возможных направлений движения робота

направлений γ_L и γ_R необходимо определить координаты точек L и R, которые принадлежат прямой (AB). С этой целью процессорные элементы PE_1 и PE_2 производят соответственно вычисление координат точек A и B:

$$X_A=R_L\cos(90°-W_L), \tag{4.36}$$

$$Y_A=R_L\sin(90°-W_L), \tag{4.37}$$

$$X_B=R_R\cos(90°-W_R), \tag{4.38}$$

$$Y_B=R_R\sin(90°-W_R). \tag{4.39}$$

Затем они определяют координаты точек L и R:

$$X_L = X_A + \frac{S_L}{R_d}(X_B - X_A), \tag{4.40}$$

$$Y_L = Y_A + \frac{S_L}{R_d}(Y_B - Y_A), \tag{4.41}$$

$$X_R = X_B + \frac{S_R}{R_d}\left(X_A - X_B\right), \qquad (4.42)$$

$$Y_R = Y_B + \frac{S_R}{R_d}\left(Y_A - Y_B\right), \qquad (4.43)$$

где R_d - ширина интервала [A B].

После этого процессорные элементы определяют направления γ_L и γ_R:

$$\gamma_L = arctg\left(\frac{X_L}{Y_L}\right) \qquad (4.44)$$

$$\gamma_R = arctg\left(\frac{X_R}{Y_R}\right) \qquad (4.45)$$

Величины S_L и S_R можно регулировать путем введения параметра D:

$$S_L(S_R)=d/2+D. \qquad (4.46)$$

Второй слой аналитического блока представляет собой нейронную сеть с прямыми связями, которая формирует совокупность возможных направлений движения в диапазоне [L R]. Два входных нейронных элемента выполняют распределительные функции. В качестве выходных нейронных элементов используются нейроны с линейной функцией активации. Количество их меняется в зависимости от ширины возможного диапазона движения:

$$n = \gamma_R-\gamma_L+1 \qquad (4.47)$$

Матрица весовых коэффициентов имеет размерность 2xn и определяется следующим образом:

$$W = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 0 & \frac{1}{\gamma_R} & \frac{2}{\gamma_R} & \dots & \frac{n-1}{\gamma_R} \end{bmatrix} \qquad (4.48)$$

Тогда выходной вектор Y представляет собой совокупность возможных направлений движения в выделенном интервале:

$$Y = W^T\gamma, \qquad (4.49)$$

где $\gamma = [\gamma_L \gamma_R]^T$

Третий слой блока предназначен для определения в интервале [L R] оптимального направления движения, которое обеспечивает минимальное угловое направление до цели (min α). Входные нейронные элементы используют конкурентный принцип функционирования. В соответствии с ним каждый нейрон вычисляет расстояние Манхэттена (the Manhattan distance) между компонентами вектораY и текущим направлением на цель α:

$$I_i = |y_i - \alpha| \qquad (4.50)$$

где $i = \overline{1,n}$.

Затем происходит определение нейрона-победителя, который обеспечивает минимальное расстояние I_i:

$$I_K = \min_i |y_i - \alpha| . \qquad (4.51)$$

Выходное значение нейрона-победителя устанавливается в единичное, а остальных нейронов - в нулевое состояние:

$$z_i = \begin{cases} 1, если\ i = k \\ 0, иначе \end{cases} \qquad (4.52)$$

В качестве входного нейрона третьего слоя блока используется нейронный элемент с линейной функцией активации. Он определяет оптимальное направление движения робота:

$$\gamma(k) = \sum_{i=1}^{n} v_i z_i = v_k z_k , \qquad (4.53)$$

где v_i - i-ая синаптическая связь линейного нейрона, k - номер нейрона-победителя.

Весовые коэффициенты v_i соответствуют компонентам вектора Y:

$$v_i = y_i, \qquad (4.54)$$

где $i = \overline{1,n}$.

Последний слой блока определения направления движения состоит из одного процессорного элемента PE_3. Он формирует оптимальное направление движения в соответствии со следующим выражением:

$$\alpha_Z = \begin{cases} \gamma(k), & \text{если } (R_L \vee R_R) > g_1 R_t \text{ или } \gamma(k) = 0 \\ \beta(k), & \text{иначе,} \end{cases}$$

где R_t – предельный порог видимости принятой системы обзора; $0 < g_1 \le 1$ – постоянный коэффициент.

Отсюда следует, что процессорный элемент PE_3 имеет два режима работы. Первый режим соответствует движению робота в следующих ситуациях:

1. Свободное пространство или тоннель, когда в нем отсутствует препятствия;

2. Если оптимальное направление γ(k) в интервале движения совпадает

с текущим направлением робота, т.е. $\gamma(k) = 0$.

Первый режим идентифицируется, когда линейные характеристики выделенного интервала движения больше величины $g_1 R_t$. Второй режим процессорного элемента PE_3 соответствует движению робота в пространстве с препятствиями, когда линейные характеристики (R_L, R_R) интервала движения робота меньше определенной величины $g_1 R_t$. Это характеризует, например, объезд одиноко стоящего препятствия или маневрирование при прохождении проемов между препятствиями. В данном режиме процессорный элемент PE_3 на основе оптимального направления $\gamma(k)$, где $k \in (AB)$, формирует оптимальное направление робота $\beta(k)$ исходя из выполнения следующих условий:

1. Обеспечение плавной траектории движения робота.

2. Направление движения робота в точке k должно быть перпендикулярно линии (AB), характеризующей расположение свободного интервала движения.

Второе условие обеспечивает исключение контакта робота при маневрах с правой или левой кромкой препятствия. Траекторией робота, удовлетворяющей перечисленным выше условиям является дуга окружности. Она должна проходить через точку K и центр робота. Центром окружности является точка C пересечения прямой (AB) и (PC), где $(PC) \perp (OK)$ и (OP)=(PK) (рис. 4.33).

Таким образом оптимальное направление робота $\beta(k)$ определяется касательной, проведенной в точке O к окружности радиусом $r = |OC|$. При этом

$$\beta(k) = b(90° + b\varphi), \tag{4.55}$$

$$b = \begin{cases} 1, & если\ x_K \geq x_c \\ -1, & если\ x_k < x_c \end{cases} \tag{4.56}$$

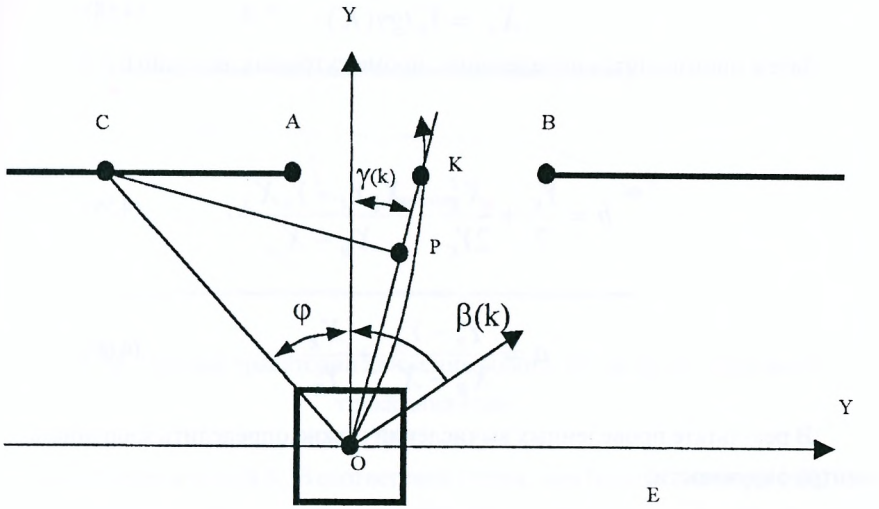


Рис. 4.33. Определение траектории движения робота: С – центр окружности; ОК – траектория движения робота соответствующая дуге окружности; ОЕ – касательная к окружности в точке О

где φ – угол, который характеризует положение центра окружности; x_k и x_c – соответственно координаты точки К и С.

Отсюда следует что для определения угла φ необходимо найти координаты точки $C(X_C, Y_C)$. Для этого процессорный элемент PE_3 вычисляет вначале координаты точки К:

$$Y_K = \frac{X_A Y_B - Y_A X_B}{(Y_B - Y_A) tg\gamma(K) - (X_B - X_A)}, \qquad (4.57)$$

$$X_K = Y_K tg\gamma(K) \cdot \tag{4.58}$$

Затем производится определение промежуточных величин b и a:

$$b = \frac{Y_K}{2} + \frac{X_K^2}{2Y_K} - \frac{X_B Y_A - Y_B X_A}{X_B - X_A}, \tag{4.59}$$

$$a = \frac{Y_B - Y_A}{X_B - X_A} + \frac{X_K}{Y_K}. \tag{4.60}$$

В результате проведенных вычислений можно определить координаты центра окружности:

$$X_C = b/a, \tag{4.61}$$

$$Y_C = \frac{Y_K}{2} + \frac{X_K^2}{2Y_K} - \frac{X_K}{Y_K} X_C. \tag{4.62}$$

На основе координат точки C процессорный элемент PE_3 вычисляет оптимальное направление робота b(k):

$$\varphi = arctg \frac{X_C}{Y_C}, \tag{4.63}$$

$$\beta(k) = b(90^\circ + b\varphi) \cdot \tag{4.64}$$

Для избежания поворотов робота на большую угловую величину, значение угла ($\beta(k)$-$\gamma(k)$) может определяться в зависимости от расстояния

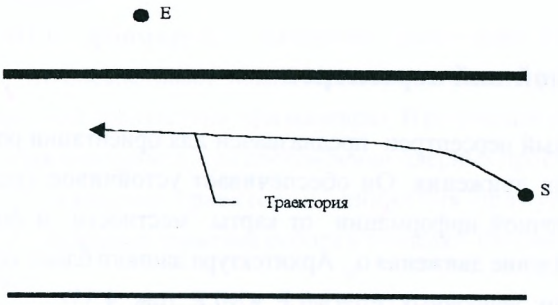


Рис. 4.34. Пример траектории движения робота: E – цель, S – начальная точка движения

между роботом и точкой K. В соответсвии с этим, чем больше значение $|OK|$, тем меньше должен быть угол $(\beta(k)-\gamma(k))$. Следует также отметить, что если интервал движения робота является свободным, т. е. $\gamma_L = \gamma_R = 90°$, то $\alpha_Z = \gamma(k)$.

Приведенный выше подход обеспечивает устойчивое движение робота в пространстве. Это происходит за счет того, что цель притягивает робота, а препятствие его отталкивает. В результате устанавливается состояние равновесия, что обеспечиваст, например, прямолинейное движение робота в коридоре (рис. 4.34).

Таким образом рассмотренный выше блок на основе аналитического подхода формирует оптимальное направление движения. Он состоит из процессорных элементов и нейронных сетей с фиксированными синаптическими связями. При этом нейронные сети являются динамическими, где количество нейронных элементов изменяется в зависимости от ширины выделенного интервала движения. Недостатком данного блока является то, что при неточной карте местности он не обеспечивает устойчивого управления роботом на узких интервалах

движения.

4.2.6. Многослойный персептрон

Многослойный персептрон предназначен для ориентации робота на узких интервалах движения. Он обеспечивает устойчивое управление роботом при неточной информации от карты местности и формирует робастное направление движения α_r. Архитектура данного блока состоит из двух многослойных нейронных сетей MLP_1 и MLP_2 (рис. 4.35)

Робастное направление движения робота a_r формируется арбитром в соответствии со следующей функцией:

$$\alpha_r = \begin{cases} \alpha_1, \ \textit{если} \ (R_L \vee R_R) \le g_1 R_t \\ \alpha_2, \ \textit{если} \ (R_L \wedge R_R) > g_1 R_t, \end{cases} \qquad (4.65)$$

где $0 < g_1 \le 1$ – постоянный коэффициент; R_t – предельный порог видимости принятой системы обзора. В соответствии с выражением (4.65) $\alpha_r = \alpha_1$ при прохождении роботом проемов между препятствиями, а $\alpha_r = \alpha_2$ при

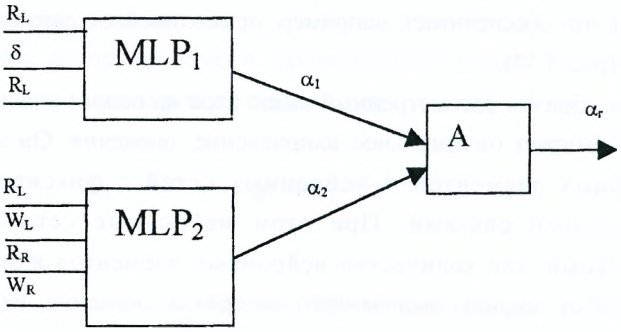


Рис. 4. 35. Архитектура многослойного персептрона: А – арбитр

движении в тоннеле.

Блок MLP_1 формирует в качестве траектории движения дугу окружности, что обеспечивает исключение столкновения робота с правой или левой кромкой препятствия при маневрах. В результате осуществляется устойчивое движение робота при прохождении дверных проемов. Блок MLP_2 использует в качестве траектории движения прямую линию, что обеспечивает устойчивое движение робота в тоннелях. Рассмотрим структуру данных блоков.

Блок MLP_1 представляет собой аналоговую трехслойную нейронную сеть с прямыми связями (рис. 4.36).

Она состоит из трех входных, восьми промежуточных и одного выходного нейрона. В качестве входной информации используются линейные (R_L и R_R) и угловые (δ) характеристики выделенного интервала движения, где $\delta=W_L+W_R$.

Перед поступлением на вход сети данные масштабируются в отрезок [0, 1] по следующим правилам:

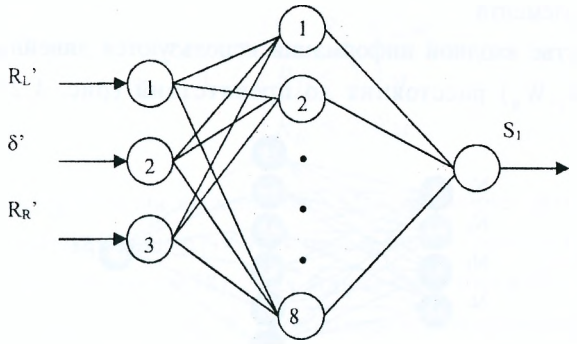


Рис. 4.36 Структура блока MLP_1

$$R'_L = R_L / 600, \tag{4.66}$$

$$R'_R = R_R / 600 \tag{4.67}$$

$$\delta' = (\delta / 100 + 1) / 2 \tag{4.68}$$

Информация на выходе нейронной сети характеризуст направление движения робота α_1. Оно получается преобразованием выходного значения S_1 нейронной сети следующим образом:

$$\alpha_1 = \text{int}(2S_1 - 1)100. \tag{4.69}$$

В результате выходные значения нейронной сети могут изменяться в диапазоне [-100°, 100°]. В качестве функции активации нейронных элементов используется сигмоидная функция.

Блок MLP_2 имеет архитектуру аналогичную блоку MLP_1 (рис. 4.37) и состоит из четырех входных, шести промежуточных и одного выходного нейронного элемента.

В качестве входной информации используются линейные (R_L,R_R) и угловые (W_L,W_R) расстояния до препятствий (рис. 4.25), которые

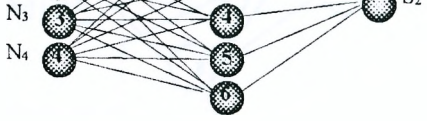


Рис. 4.37 Структура блока MLP_2

масштабируются в соответствии со следующими выражениями:

$$N_1=R_L/600, \quad (4.70)$$

$$N_2=R_R/600, \quad (4.71)$$

$$N_3=(W_L/100+1)/2, \quad (4.72)$$

$$N_4=(W_R/100+1)/2. \quad (4.73)$$

Информация на выходе нейронной сети MLP_2 характеризует направление движения робота α_2, которое определяется следующим образом:

$$\alpha_2=\text{int}(2S_2-1)100, \quad (4.74)$$

где S_2 – выходное значение нейронной сети.

Для обучения нейронных сетей MLP_1 и MLP_2 необходимо генерировать тренировочные наборы. Каждый набор представляется в числовой форме и состоит из нескольких входных и одного выходного значения. Рассматриваемый блок предназначен для ориентации робота на узких интервалах движения, ширина которых меньше двух метров. С учетом принятой системы обзора робота радиусом R_1 = 2.4 метра получается следующая область V, в которой необходимо генерировать тренировочные наборы:

$$V \in \begin{cases} 1 < R_d \le 2м \\ R_L \le 2.4м \\ R_R \le 2.4м. \end{cases} \quad (4.75)$$

В качестве траектории движения блок MLP_1 формирует дугу окружности, которая проходит через центр робота и определенную точку К в выделенном интервале движения (рис 9.38). Зная координаты точки К и координаты интервала движения (X_A, Y_A, X_B, Y_B) можно определить траекторию движения робота и направление движения в каждой точке (раздел 4.2.5). В результате для одного положения робота относительно выделенного промежутка движения

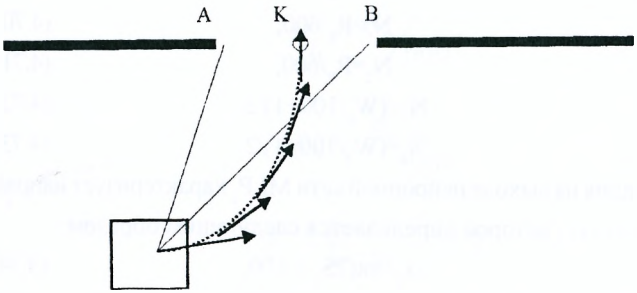


Рис 4.38 Траектория движения робота, соответствующая дуге окружности

получается совокупность тренировочных наборов.

Производя вращение выделенного интервала движения [A B] и точки K относительно центра робота (рис 4.39) можно получить различные тренировочные наборы. Изменяя в области V расположение робота относительно интервала движения и выполняя перечисленные выше операции можно получить обучающую выборку, которая состоит из

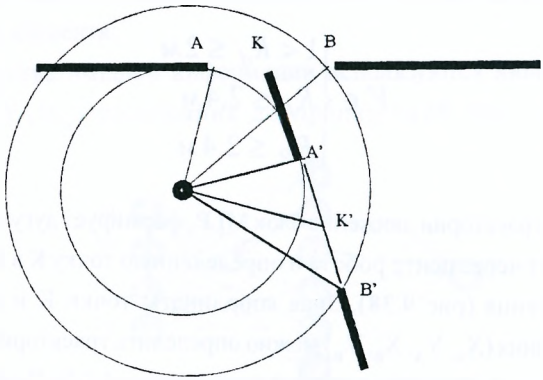


Рис 4.39 Формирование обучающей выборки путем вращения интервала движения [AB]

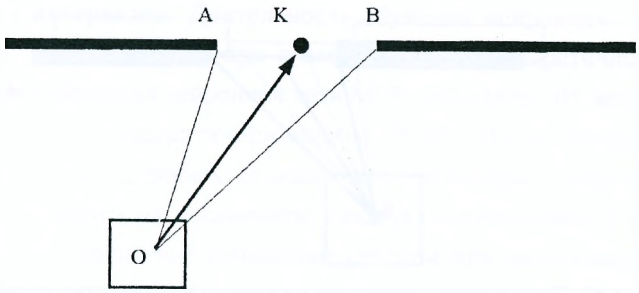


Рис 4.40 Текущее направление движения OK формируемое блоком MLP_2

множества тренировочных наборов

Аналогичным образом производится формирование обучающей выборки для блока MLP_2. Здесь в качестве текущего направления движения робота выбирается точка K (рис. 4.40).

Изменяя в области V положение робота и вращая его с выбранным шагом дискретности относительно точки O (рис. 4.40), получается совокупность тренировочных наборов.

В случае неточной информации от карты местности реальное положение препятствия может отличаться от того, что видит робот. Такая ситуация показана на рис 4.41, где сплошными линиями изображено положение препятствия, которое видит робот, а пунктирными – реальное положение препятствия.

Для обеспечения робастного управления роботом при неточной карте местности необходимо соответствующим образом выбирать положение точки K в выделенном интервале движения. Тогда, если обучить нейронную сеть правильным выходным данным, то она сможет обеспечить устойчивое управление роботом при неточной карте местности. Концепция обучения нейронных сетей MLP_1 и MLP_2 в общем случае состоит из следующих шагов:

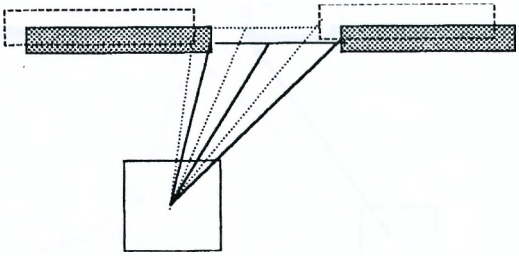


Рис. 4.41. Возможное искажение видимого промежутка движения

- Оператор управляет роботом, моделируя при этом прохождение различных по ширине интервалов движения.
- Для каждого интервала движения определяется точка К, характеризующая реальное положение робота в этом интервале и характеристики интервала (W_L, W_R, R_L, R_R) при различных положениях робота относительно его.
- Путем определения траектории движения и вращения входных и выходных данных, как было показано выше в диапазоне 180°, формируются обучающие выборки.
- Производится обучение нейронных сетей MLP_1 и MLP_2 методом обратного распространения ошибки.

Данный алгоритм характеризуется минимальным набором экспериментальных данных. Достаточно определить только положение точки К и характеристики интервала движения. Проведено компьютерное моделирование многослойного персептрона. Объем обучающей выборки для блока MLP_1 составил 92 тренировочных набора, а для блока MLP_2 – 60 наборов.

При этом для генерации обучающих наборов не использовалось вращение информации, а варьировалось положение робота относительно

различных интервалов. Длительность обучения нейронной сети MLP_1 составила $5*10^3$ итераций при суммарной среднеквадратичной ошибке $5*10^{-5}$. Время обучения нейронной сети MLP_2 составила 10^4 итераций при суммарной среднеквадратичной ошибке $5*10^{-5}$. После обучения робот успешно проходил с любых позиций через различные узкие интервалы движения, несмотря на искаженную входную информацию. В результате обеспечивается робастное управление роботом при недостоверной карте местности.

4.2.7. Бинарный блок

Недостатком работы предыдущих блоков определения направления движения состоит в том, что они не учитывают расстояния с боковых сторон робота до препятствия. В результате при осуществлении роботом маневров может происходить столкновение с препятствиями. Такая ситуация показана на рис. 4.42, когда при повороте налево происходит контактирование робота с

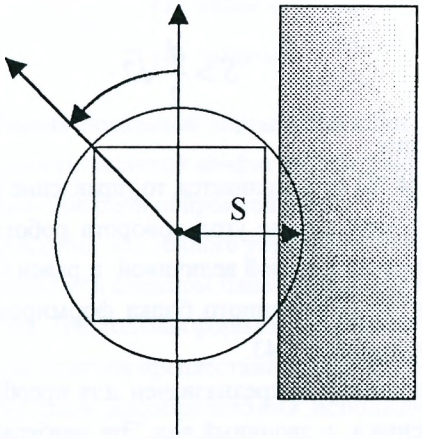


Рис. 4.42 Столкновение робота с препятствием при повороте налево

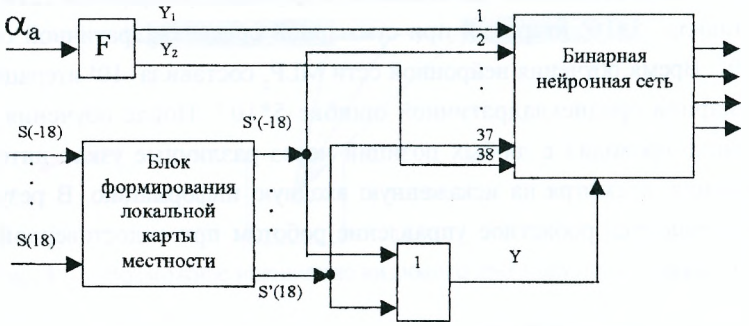


Рис. 4.43 Структурная схема бинарного блока

препятствием.

Для осуществления подобных маневров необходимо, чтобы боковое расстояние до препятствия было больше радиуса описанной вокруг робота окружности:

$$S > \frac{d}{2}\sqrt{2}, \qquad (4.76)$$

где d – ширина робота.

Если условие (4.76) не выполняется, то управление роботом производит бинарный блок. В этом случае угол поворота робота в том или ином направлении является постоянной величиной и равен одному градусу.

Структурная схема бинарного блока формирования направления движения изображена на рис. 4.43.

Блок преобразования F предназначен для преобразования углового направления движения a_a в двоичный вид. Это необходимо для управления бинарной нейронной сетью. Блок F выполняет следующие функции:

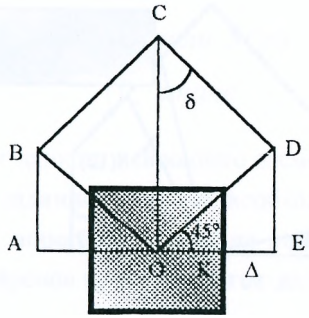


Рис. 4.44 Конфигурация локальной карты местности

$$Y_1 = \begin{cases} 1, & \textit{если } \alpha_a > 0 \\ 0, & \textit{иначе} \end{cases} \qquad (4.77)$$

$$Y_2 = \begin{cases} 1, & \textit{если } \alpha_a < 0 \\ 0, & \textit{иначе} \end{cases} \qquad (4.78)$$

Блок формирования локальной карты местности предназначен для получения карты местности заданной конфигурации (рис. 4.44) и генерации сигнала Y возбуждения бинарной нейронной сети.

Такая карта необходима для точного управления роботом в ситуациях, когда препятствие находиться слишком близко (на расстоянии меньше D) с боковой стороны робота. Треугольная форма выбрана исходя из обеспечения плавных маневров при наличии препятствий по фронту робота. В качестве входной информации блок преобразования использует сжатую карту местности, состоящую из 36 элементов. Технология преобразования состоит в том, что если препятствие находиться в зоне ABCDE, то соответствующие

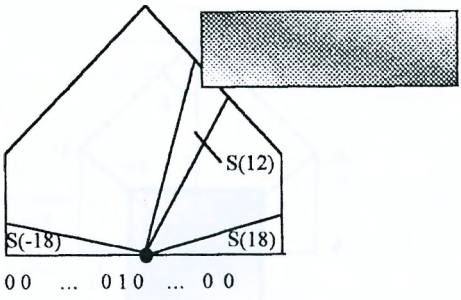


Рис. 4.45 Пример формирования локальной карты местности

элементы S'(p) устанавливаются в единичные значения, в противном случае в нулевые значения (рис. 4.45)

В результате получается бинарный массив, который характеризует наличие в заданной области препятствий. Структура блока преобразования представляет собой один слой пороговых нейронных элементов (рис. 4.46), каждый из которых соответствует определенному сектору карты местности.

Нейронные элементы выполняют следующие функции:

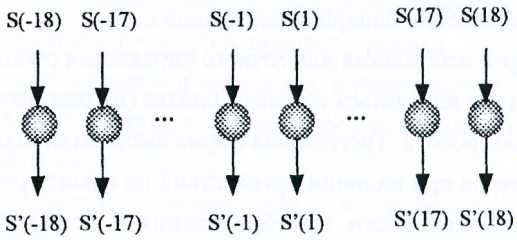


Рис. 4.46 Структура блока формирования локальной карты местности

$$S'(p) = \begin{cases} 1, & \textit{если } S(p) \le T(p) \\ 0, & \textit{иначе} \end{cases} \qquad (4.79)$$

Здесь T(p) – порог данного нейронного элемента. Для формирования бинарного массива в заданной области необходимо соответствующим образом формировать пороговые значения нейронов. В области ODE пороговые значения нейронов представляются как:

$$T(p) = \frac{d/2 + \Delta}{\cos\alpha(p)}, \qquad (4.80)$$

где p=18÷10, d – ширина робота, а D=|KE|.

Аналогичным образом рассчитываются пороги в области OAB, где p= -18÷10. Величина D характеризует минимальные боковые расстояния до препятствия, при котором возможно осуществление различных маневров:

$$\Delta = \frac{d}{2}(\sqrt{2} - 1). \qquad (4.81)$$

На каждую из рассмотренных областей приходиться по 9 нейронов (S'(-18)…S'(-10),S'(18)…S'(10)). Эти нейроны участвуют в формировании сигнала возбуждения бинарной нейронной сети (рис. 4.47).

В результате, если препятствие находиться в зоне OAB или ODE, то бинарная сеть производит управление роботом. В противном случае управление осуществляется от блоков 3 и 4. В области OBCD пороговые значения формируются следующим образом:

$$T(p) = \frac{\sin\delta}{\sin\gamma}h, \qquad (4.82)$$

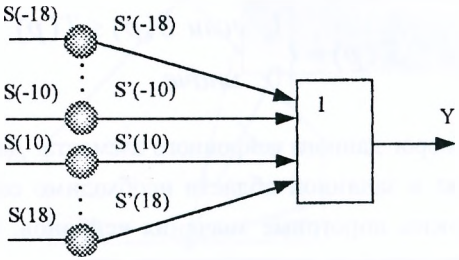


Рис. 4.47 Формирование сигнала возбуждения Y.

$$\gamma = 180° - \delta - \alpha(p), \tag{4.83}$$

$$\sin\delta = \frac{d/2 + \Delta}{\sqrt{(d/2 + \Delta)^2 + (h - d/2 - \Delta)^2}}, \tag{4.84}$$

где h=|OC|.

Нейронные элементы с рассчитанными таким образом пороговыми значениями будут формировать карту местности заданной конфигурации (рис. 4.44). Бинарная нейронная сеть предназначена для управления роботом, когда повороты на большие значения могут привести к столкновению с препятствием. В этом случае Y=1. Такая сеть представляет собой трехслойную нейронную сеть с прямыми связями. (рис. 4.48).

В качестве функции активации нейронных элементов используется сигмоидная функция. На выходе нейронной сети формируются команды управления роботом (рис. 4.49).

При этом поворот в определенный момент времени происходит на 1°, что исключает столкновение робота с боковыми препятствиями. Для

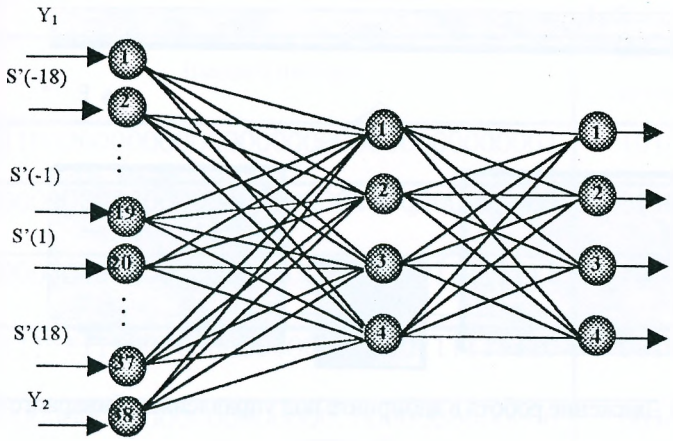


Рис. 4.48 Бинарная нейронная сеть.

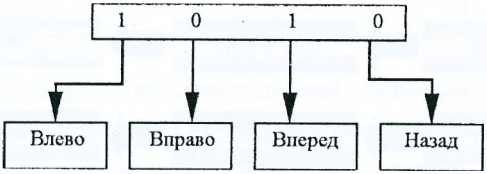


Рис. 4.49 Команды управления роботом

управления бинарной сетью используются также сигналы Y_1 и Y_2, которые получаются путем преобразования направления движения α_a от блока 5 (рис.4.17). Так, если Y_1=1, что соответствует α_a>0, то бинарная сеть будет формировать команду поворота направо на величину 1°. Такое взаимодействие блоков 3,4 и 5 обеспечивает движение робота в ближайшем направлении к цели. Особенно актуальным это является при существовании альтернативных маршрутов движения в узких лабиринтах (рис. 4.50).

Бинарная сеть работает по принципу огибания препятствия. Возможные варианты ее функционирования изображены на рис. 4.51

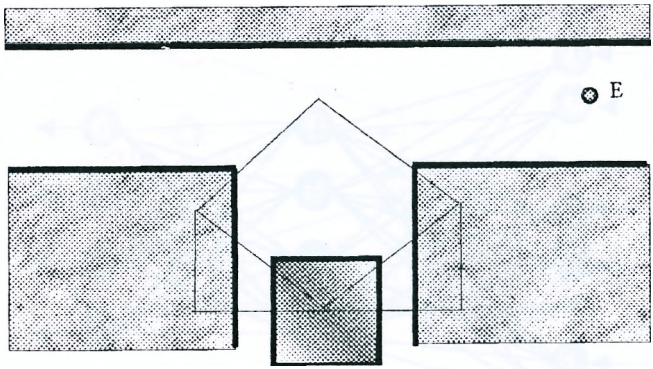


Рис. 4.50 Движение робота в лабиринте под управлением бинарного блока: E – цель

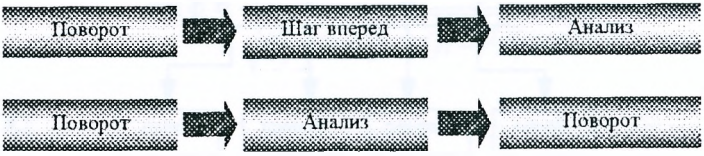


Рис. 4.51Формирование команд управления бинарной нейронной сетью

Во втором варианте поворот происходит до тех пор пока не произойдет огибание препятствия.

Для обучения бинарной сети необходимо сформировать тренировочные наборы. Генерация обучающей выборки характеризуется простотой и вытекает из логики работы данной сети. В табл. 4.1 приведены тренировочные наборы для некоторых ситуаций.

Для обучения бинарной сети использовался алгоритм обратного распространения ошибки. Объем обучающей выборки составил 30 наборов. Длительность обучения – $3*10^3$ итераций, суммарная среднеквадратичная

Таблица 4.1

Входной паттерн	Выходной паттерн
01100000000000000000000000000001	1010
10000000000000000000000000000000	0110
00000000000000000000000111111111111	1000
01111111100000000000000000111111110	0010
00000000000000000000000000000000000	0010
11111111111111111111111111111111111	0001

ошибка – $1*10^{-3}$. Наряду с логическим подходим к формированию обучающей выборки может применяться алгоритм рассмотренный в разделе 4.2.6, который основан на моделировании различных ситуаций оператором при ручном управлении роботом. Это обеспечивает формирование корректного направления движения при неточной карте местности.

4.2.8. Самообучение и самоорганизация

Самообучение характеризуется способностью системы обучаться при взаимодействии с внешней средой. В результате самообучения происходит самоорганизация системы с целью адаптации к внешней среде. Самообучение позволяет освободить оператора от процесса обучения и является важным фактором для эволюции системы. Так, при функционировании робота в разных условиях точность информации от

сенсорных устройств может быть различной. В результате возникает необходимость корректировать знания, заложенные в систему, с целью адаптации к внешней среде. Особенно актуальным это является при функционировании робота в агрессивных средах или на других планетах, где невозможно предусмотреть все аспекты ситуационного взаимодействия робота с окружающей обстановкой. Рассмотрим основные принципы реализации концепции самообучения для мобильных роботов.

Общий подход к построению самообучающейся системы состоит в том, что начальные знания робота могут пополняться и корректироваться в процессе функционирования. При этом здесь предполагается, что базовые знания робота содержатся в блоках 1–3 и 6 (рис. 4.17), которые определяются логическим путем, как было показано в предыдущих разделах. Тогда задача состоит в том, чтобы в процессе функционирования робота обучить многослойный персептрон (блок 4) для обеспечения робастного управления на узких интервалах движения. Схема взаимодействия робота с внешней средой в процессе самообучения представлена на рис. 4.52.

Управление при этом происходит от аналитического блока и бинарной нейронной сети. Процесс самообучения происходит методом проб и ошибок на узких интервалах движения. При успешном выполнении маневра формируются тренировочные наборы для обучения многослойного персептрона. При неудачной попытке происходит возвращение робота в исходную точку на несколько шагов назад и повторение маневра (рис. 4.53).

Блок анализа ситуаций (рис. 4.52) предназначен для восстановления ситуации на предыдущем шаге робота t-1 и формирования корректирующего направления движения $\gamma(k_1)$:

$$\gamma(k_1) = \gamma(k) \pm \delta$$

где $\gamma(k)$ – направление движения, сформированное аналитическим блоком в данной точке на предыдущей попытке маневра; δ – угол коррекции

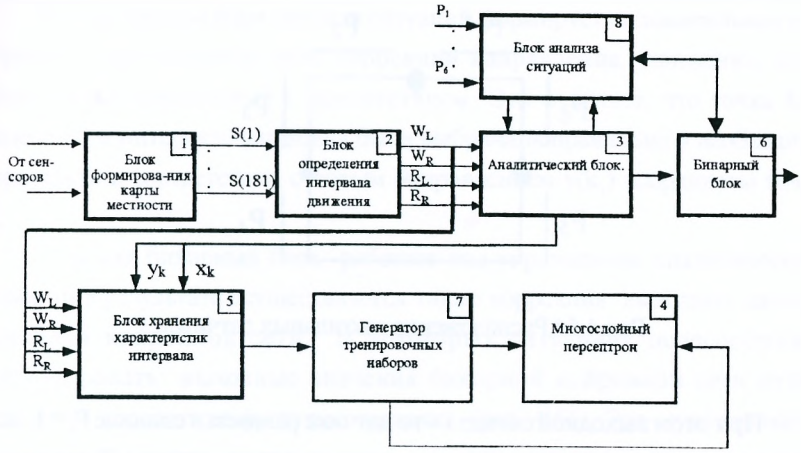


Рис 4.52. Архитектура нейронной системы в режиме самообучения

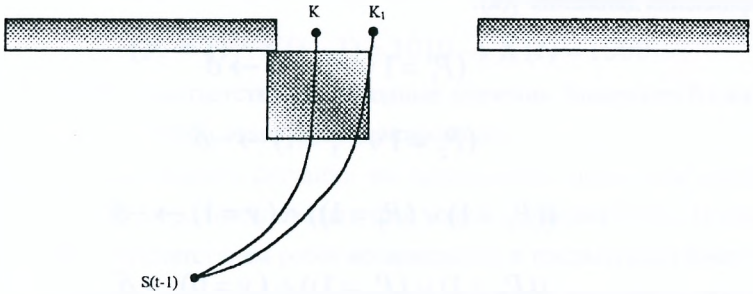


Рис. 4.53. Пример некорректной траектории робота (точка K): K_1 – скорректированное направление движения

направления движения.

В качестве входной информации блока анализа ситуации используются данные от тактильных датчиков (рис. 4.53).

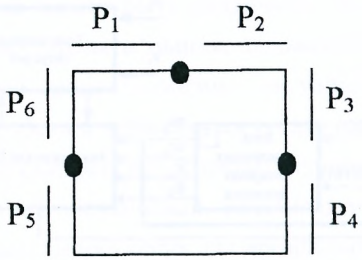


Рис.4.54 Расположение тактильных датчиков.

При этом выходной сигнал i – го датчика равняется единице $P_i = 1$, если произошел контакт соответствующего датчика с препятствием. В противном случае $P_i = 0$. Коррекция направления движения робота производится путем логического анализа информации от тактильных датчиков и предыдущего направления движения $\gamma(k)$:

$$(P_1 = 1 \vee P_6 = 1) \rightarrow \delta$$

$$(P_2 = 1 \vee P_3 = 1) \rightarrow -\delta$$

$$((P_4 = 1) \vee (P_5 = 1)) \wedge (y = 1) \rightarrow -\delta$$

$$((P_4 = 1) \vee (P_5 = 1)) \wedge (y = 0) \rightarrow \delta$$

В приведенных выше выражениях сигнал y формируется следующим образом:

$$y = \begin{cases} 1, & \textit{если } \gamma(k) > 0 \\ 0, & \textit{иначе} \end{cases}$$

Таким образом блок анализа ситуаций формирует положительное или отрицательное значение угла коррекции направления движения, если произошло столкновение с препятствием. Это означает, что точка K в выделенном интервале движения была выбрана неправильно и необходимо определить в соответствии с новым направлением $\gamma(k_1)$ координаты точки K_1.

Так как бинарный блок работает под управлением аналитического блока, то в результате осуществляется также коррекция выходных данных бинарной нейронной сети. В некоторых ситуациях целесообразно корректировать выходные значения бинарной нейронной сети путем изменения варианта ее функционирования (рис. 4.51). Это осуществляется путем анализа информации от тактильных датчиков и предыдущих выходных значений бинарного блока. Например, если

$$(P_1 = 1) \wedge K(t-1) = 0110 \rightarrow K(t) = 0100,$$

$$(P_2 = 1) \wedge K(t-1) = 1010 \rightarrow K(t) = 1000,$$

где K(t-1) и K(t) – соответственно выходные значения бинарного блока на предыдущем и текущем этапе функционирования.

Для восстановления ситуации на предыдущем шаге блок анализа ситуации хранит координаты предыдущей точки движения S(t-1). В случае столкновения с препятствием робот возвращается в предыдущую точку S(t-1). Блок хранения характеристик интервала содержит координаты текущей точки K и расположение интервала движения (W_L,W_R,R_L,R_R). В случае успешного завершения маневра роботом они поступают в генератор тренировочных наборов. Маневр считается успешным, если робот достигает точки K в выделенном интервале движения без столкновений с препятствиями. В этом случае координаты точки K относительно подвижной

системы координат робота равняются нулю.

Генератор на основе координат точки K и характеристик интервала движения путем вращения (раздел 4.2.6) определяет для блоков MLP_1 и MLP_2 (рис. 4.35) совокупность тренировочных наборов, количество которых равняется приблизительно 30. В результате моделирования различных ситуаций формируется обучающая выборка. Как показали эксперименты для устойчивой работы аналоговых нейронных сетей MLP_1 и MLP_2 необходимый объем обучающей выборки составляет 120 – 240 тренировочных наборов. В процессе функционирования робота в случае необходимости осуществляется также коррекция обучающей выборки для бинарного блока. Для устойчивой работы его достаточно 40 тренировочных наборов.

Применение бинарного блока для управления роботом в режиме самообучения позволяет уменьшить количество ошибок при совершении маневров и следовательно ускорить процесс самообучения. При этом самообучение может происходить как для получения новых, так и для коррекции старых знаний. В результате этого обеспечивается адаптация робота к внешней среде.

4.2.9. Тестирование

Тестирование нейронной системы осуществлялось как при помощи компьютерного моделирования движения робота, так и при проведении натурных экспериментов.

Натурные эксперименты проводились на роботе 'Walter' в 1997 году. В качестве полигона для движения использовалась лаборатория робототехники и коридор здания. Задача робота состояла в достижении конечной точки движения при ориентации на незнакомой местности с препятствиями.

Для автономного управления роботом использовалась нейронная система состоящая из первого второго и третьего блоков (рис. 4.17).

Рис. 4.55. Изображение проема двери, полученное в лаборатории робототехники (Германия) при помощи видеокамеры, установленной на роботе "Walter"

Эксперименты показали довольно устойчивое движение робота в незнакомом пространстве. Проблемы возникали в основном при прохождении узких интервалов движения, таких как проем двери (рис. 4.54), ширина которого равнялась 100см.

При этом в окрестности проема находилось различное лабораторное оборудование, что создавало дополнительные помехи. Задача робота состояла в том, чтобы с различных позиций преодолеть дверной проем. На начальном этапе экспериментов робот проезжал проем в меньшинстве своих попыток. Анализ карты местности показал , что та генерирует искаженную ширину дверного проема или смещение его относительно реального положения. Путем регулирования параметра Δ (раздел 4.2.5) удалось достичь приблизительно 90% успешных попыток прохождения робота через дверной проем. Натурные эксперименты явились основой для доработки нейронной

системы с целью обеспечения робастного управления.

В результате система была доработана в соответствии со схемой изображенной на рис.4.17. Проведено программное моделирование такой системы при движении робота в различных условиях.

Программное моделирование показало устойчивое функционирование робота на различных участках движения. На рис. 4.55 изображена траектория робота при движении его в лабиринте.

Проведены также эксперименты при движении робота на узких интервалах движения, таких как дверной проем. При этом моделирование и обучение проводилось при неточной информации от карты местности, когда расположение препятствия отличается от того, что видит робот. На рис. 4.56 показано прохождение робота через проем под управлением блока 4 в условиях неточной информации.

Как следует из рисунка обучение нейронной сети на корректные выходные данные нейтрализует искаженную входную информацию. За счет

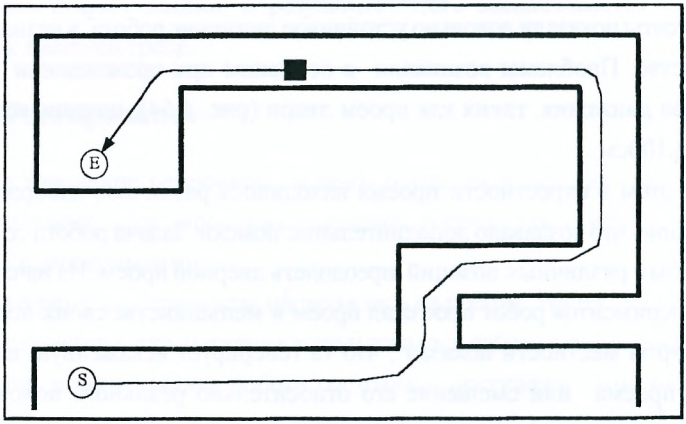


Рис. 4.56. Движение робота в лабиринте: S – начальная точка движения; E – конечная

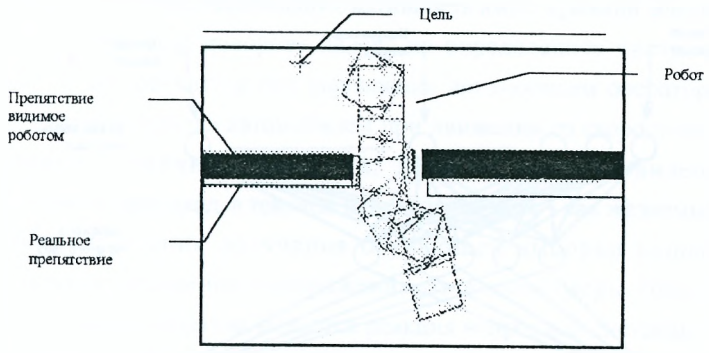


Рис. 4.57. Движение робота к цели при неточной карте местности

этого обеспечивается робастное управление роботом. Осуществлены компьютерные эксперименты по самообучению нейронной системы, в результате которых происходило обучение многослойного персептрона и бинарного блока при взаимодействии робота с внешней средой. При этом моделировалась недостоверная информация от карты местности. После обучения робот продемонстрировал устойчивое функционирование на различных участках движения.

4.3. Автономное управление автомобилем

Нейронные сети могут эффективно использоваться для управления автомобилем. В этом случае достаточно задать системе управления координаты конечной точки движения и автомобиль без участия человека будет двигаться к цели. Такая система разработана в университете Карнеги-Меллона в рамках ALVINN проекта (Autonomous Land Vehicle In a Neural Networks) [40,41]. В ней предполагается, что автомобиль оборудован видеокамерой, которая отображает дорогу с разметкой. Центральным

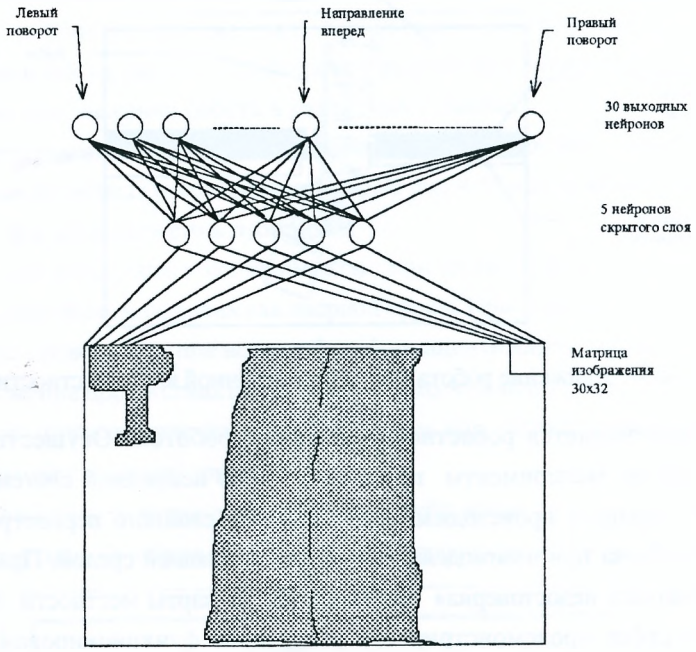


Рис. 4.58. Отображение изображения дороги на нейронную сеть

элементом такой системы является трехслойная нейронная сеть с прямыми связями (рис. 4.57).

Входной слой содержит 30x32 нейронных элемента на которые подается преобразованное от видеокамеры изображение пути. Скрытый слой состоит из пяти, а выходной слой из 30-ти нейронных элементов. В качестве функции активации используется сигмоидная функция. Активность выходных нейронов характеризует поворот руля в ту или иную сторону. Так, если максимальной активностью обладает центральный нейрон, то это означает

движение прямо. Когда наибольшую активность имеет крайний левый нейрон, то это соответствует повороту налево на определенное число градусов. Нейронная сеть обучается при управлении автомобилем оператором. При этом оператор управлял автомобилем при движении со скоростью 9.5 км/ч моделируя различные ситуации. Изображение от видеокамеры использовалось как вход, а текущее направление руля – как желаемый выход. С целью упрощения получения обучающей выборки используется программное вращение изображения от видеокамеры (рис. 4.58) и соответствующим образом меняется реакция нейронной системы.

В результате была создана обучающая выборка, объем которой составил 1200 тренировочных наборов. Обучение нейронной сети проводилось с

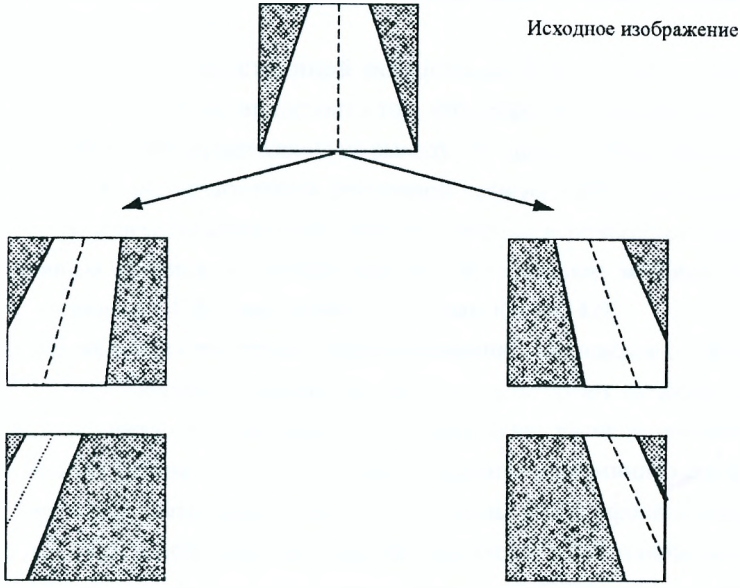


Рис. 4.59. Вращение исходного изображения от видеокамеры

использованием трех станций «Sun-4». Время обучения методом обратного распространения ошибки составило пять минут. После обучения, как показали эксперименты, нейронная сеть может автономно управлять автомобилем. В настоящее время в рамках этого проекта была достигнута скорость движения автомобиля до 70 миль/ч. При этом автомобиль проехал 90 миль к северу от Питтсбурга. Это говорит о большом потенциале нейронных сетей для решения различных задач.

ГЛАВА 5. ОТОБРАЖЕНИЯ НЕЙРОННЫХ СЕТЕЙ НА СИСТОЛИЧЕСКИЕ ПРОЦЕССОРЫ

Нейронные сети характеризуются параллельной архитектурой и соответственно параллельной обработкой информации. Эмуляция нейронных сетей на стандартных микропроцессорах приводит к неэффективности вычислений, что связано с последовательным, функционированием микропроцессора и противоречит параллельной природе нейронных сетей. Поэтому для моделирования нейронных сетей на системном уровне используются различные многопроцессорные конфигурации, например, транспьютерные сети [14].

Проблема непосредственной реализации нейронных сетей на микроэлектронной технологии состоит в том, что нейронные элементы имеют большой коэффициент разветвления по выходу. Это делает затруднительным прямое отображение архитектуры нейронной сети на СБИС-технологию. Поэтому для реализации нейронных сетей на микроэлектронной технологии перспективным является их отображение на систолические массивы. Они были предложены в 1978 г. американским ученым Куном [42].

Систолический - это термин, характеризующий сокращение сердечной мышцы. В систолическом процессоре данные пульсируют подобно току крови в теле человека. Основной принцип систолических вычислений состоит в том, чтобы все данные будучи регулярно подкачаны и ритмично переданы по массиву, могли быть эффективно использованы всеми процессорными элементами [42]. Систолический массив характеризуется однородными элементами, локальными и регулярными связями, а также пространственным и временным параллелизмом. Это позволяет легко реализовать систолические процессоры на микроэлектронной технологии. Таким образом систолические

массивы позволяют соединить преимущества конвейерной обработки информации и технологии СБИС. Рассмотрим отображение различного рода нейронных сетей на систолические массивы.

5.1. Однослойные нейронные сети

В данном разделе описывается реализация однослойных нейронных сетей на одномерных и двухмерных систолических структурах. Они представляют собой совокупность процессорных элементов соединенных локальными связями.

5.1.1. Линейные систолические массивы

Рассмотрим нейронную сеть, состоящую из одного слоя обрабатывающих элементов (рис. 5.1)

Выходные значения сети можно представить в виде нелинейного преобразования взвешенной суммы:

$$Y=F(S)$$

Взвешенная сумма определяется как произведение матрицы весовых

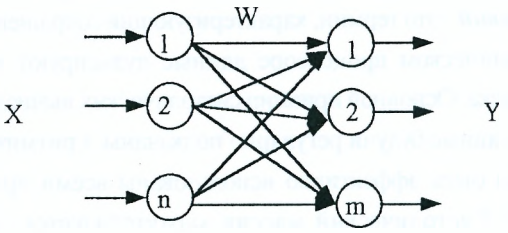


Рис. 5.1. Однослойная нейронная сеть

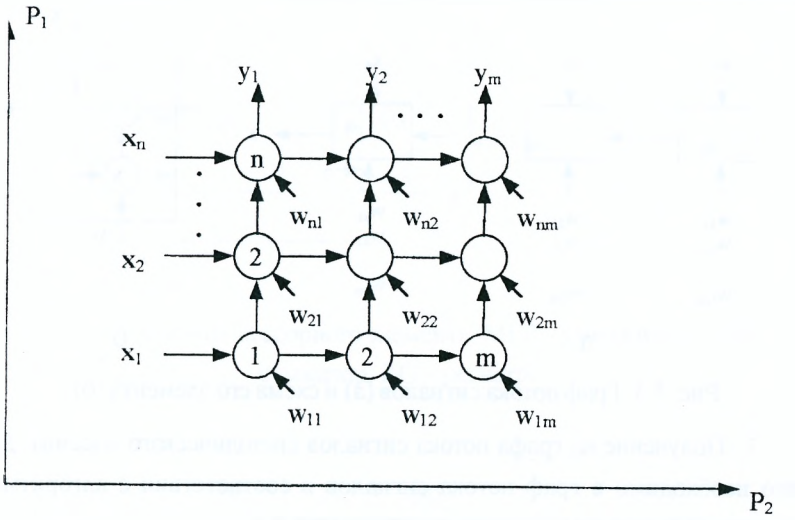


Рис 5.2. Граф зависимостей

коэффициентов на вектор входных сигналов:

$$S=W^{T}X$$

В работе [108] предложена формальная процедура проектирования систолических процессоров. Она состоит из следующих этапов:

1. Построение графа зависимостей вычислений. Такой граф описывает вычисления представленные в алгоритме и характеризуется локальностью связей. Так граф зависимостей для однослойной нейронной сети без учета операции нелинейного преобразования представлен на рис. 5.2.

2. Отображение графа зависимостей в граф потока сигналов. Такое отображение получается проецированием графа зависимостей на одну из выбранных плоскостей. Так, при проецировании графа зависимостей на плоскость P_1 (рис.5.2) получится граф потока сигналов, изображенный на рис.5.3.

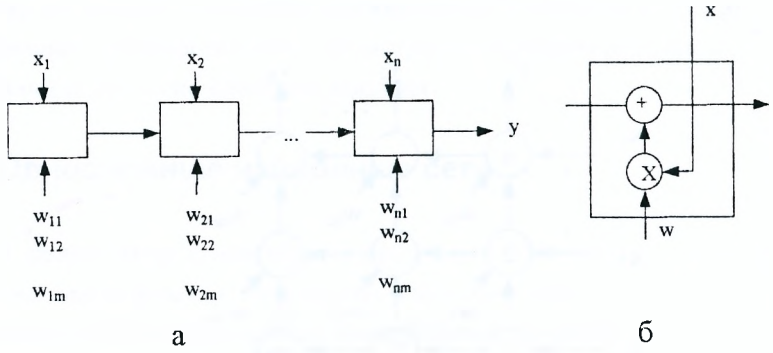


Рис. 5.3. Граф потока сигналов (а) и схема его элемента (б)

3. Получение из графа потока сигналов систолического массива. Для этого необходимо в граф потока сигналов в соответствии с алгоритмом вычислений ввести необходимые задержки (рис.5.4)

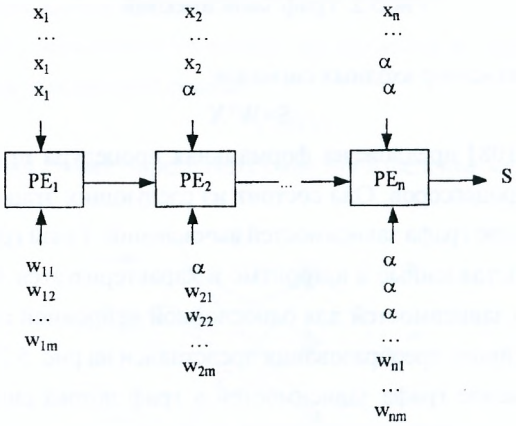


Рис. 5.4. Систолический процессор: S – взвешенная сумма; α – означает, что в данный момент времени информация не подается на соответствующий процессорный элемент

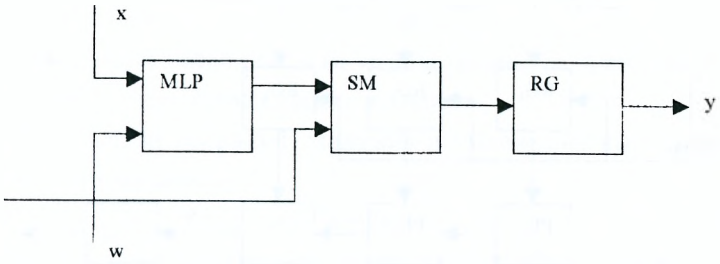


Рис. 5.5. Схема процессорного элемента: MLP – умножитель; SM – сумматор; RG – регистр

Таблица 5.1.

Номер такта	PE_1	PE_2	PE_3
1	$w_{11}x_1$	–	–
2	$w_{12}x_1$	$w_{21}x_2+w_{11}x_1$	–
3	$w_{13}x_1$	$w_{22}x_2+w_{12}x_1$	$w_{21}x_2+w_{11}x_1+w_{31}x_3$

Схема процессорного элемента (PE) представлена на рис. 5.5.

В таблице 5.1 представлено функционирование систолического массива из трех процессорных элементов для нескольких тактов.

Анализируя схему, приведенную на рис. 5.4, можно заметить, что входные данные x_i для процессорного элемента PE_i являются постоянной величиной. Для такого потока входных данных можно предложить простую схему управления поступлением информации в процессорные элементы. Преобразуя систолический массив подобным образом и добавляя в него блок нелинейного преобразования F получим окончательный вариант реализации однослойной нейронной сети, изображенной на рис.5.6.

Занесение потока входных данных $X=(x_1,x_2,x_3)$ в процессорные

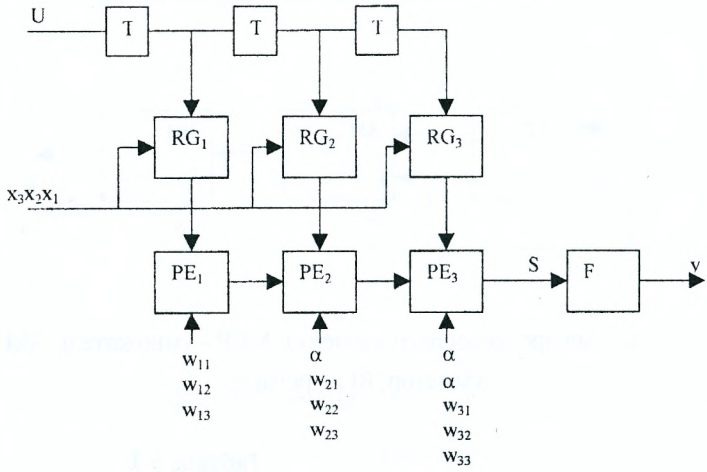


Рис.5.6. Реализация однослойной нейронной сети на систолическом процессоре для n = 3

элементы осуществляется в такой схеме под управлением цепочки триггеров (Т), которая образует сдвигающий регистр. При этом сигналы управления U цепочкой триггеров образуются следующим образом:

$$U=\{100...0\}$$

Тогда в первом такте в регистр RG_1 процессорного элемента PE_1 занесется информация x_1, во втором такте в регистр процессорного элемента PE_2-x_2 и т.д. Рассмотрим характеристики полученного выше систолического массива. Размерность его равняется размерности вектора X входных сигналов, а связи между процессорными элементами являются однонаправленными. Данные Y на выходе линейки процессорных элементов появляются друг за другом в последовательных тактах синхронизации. Определим производительность такого процессора. Пусть α-время такта работы конвейера, n - размерность вектора X, а m - размерность вектора Y.

Тогда общее время вычислений равняется:

$$t_0=(n+Lm)\alpha,$$

где n характеризует число тактов на загрузку конвейера, а m - число тактов вычислений, L - количество входных паттернов.

Общее число операций сложения и умножения выполняемых в систолическом массиве

$$V=(2n-1)mL$$

Тогда производительность процессора в общем режиме можно определить как

$$\gamma_1 = \frac{V}{t_0} = \frac{(2n-1)mL}{(n+Lm)\alpha}$$

Производительность в режиме насыщения (без учета загрузки конвейера) равняется:

$$\gamma_n = \frac{2n-1}{\alpha}$$

Рассмотрим другие варианты систолического процессора для реализации однослойной нейронной сети. При проецировании графа зависимостей на плоскость P_1 получается систолический массив, изображенный на рис.5.7.

Схема процессорного элемента представлена на рис.5.8.

Такой систолический процессор характеризуется однонаправленными связями. Выходное значение y_j снимается с j-го процессорного элемента. Каждый процессорный элемент соответствует нейронному элементу. Размерность систолического массива соответствует размерности вектора выходных сигналов Y. Поэтому с точки зрения аппаратных затрат, если n>m,

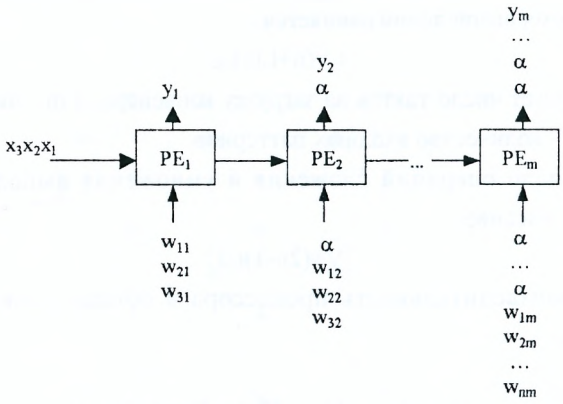


Рис. 5.7. Линейный систолический массив

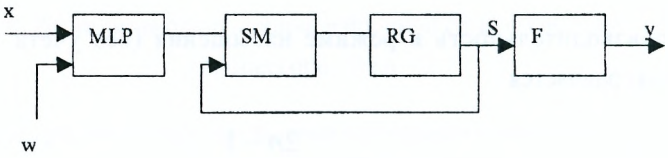


Рис.5.8. Схема процессорного элемента: F – блок нелинейного преобразования

то выгодно использовать данный вариант организации систолического массива, а при n<m целесообразно использовать предыдущий вариант.

С точки зрения производительности, первый вариант является предпочтительнее, так как данные на выходе его всегда появляются в каждом такте конвейера. Во втором варианте при смене входных данных X, выходные данные будут появляться соответственно через n, n+1 и т.д. тактов. Производительность такого процессора в общем режиме равняется:

$$\gamma_2 = \frac{(2n-1)m}{(n+m)\alpha}$$

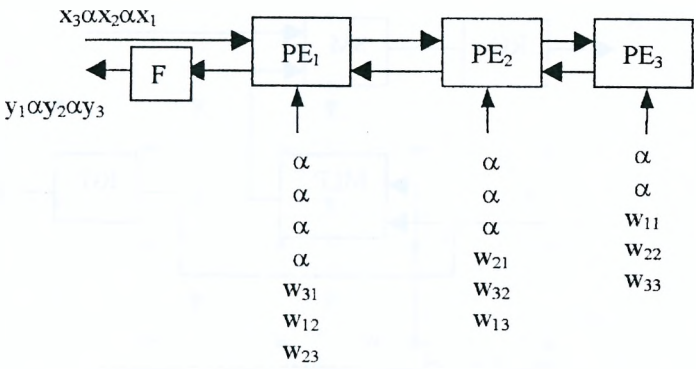


Рис. 5.9. Линейный систолический массив с разнонаправленными связями

Очевидно, что $\gamma_1 > \gamma_2$.

Рассмотрим реализацию однослойной нейронной сети на систолическом массиве с разнонаправленными связями. Схема такого процессора показана на рис. 5.9.

Схема процессорного элемента для систолического массива с разнонаправленными связями изображена на рис.5.10.

Такой систолический массив характеризуется движением входных и выходных данных в противоположных направлениях. При этом входная и выходная информация поступает через граничный процессорный элемент. Размерность систолического процессора равняется размерности n вектора X.

Определим производительность приведенного выше систолического массива. Общее время вычислений равняется

$$t_0 = 2\alpha(n+Lm-1)$$

Тогда производительность процессора в общем режиме равняется

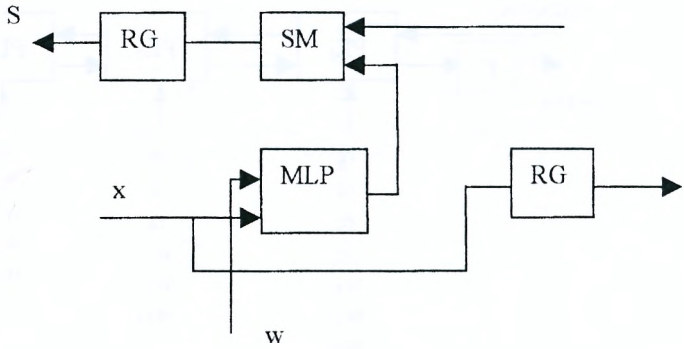


Рис.5.10. Схема процессорного элемента

$$\gamma_3 = \frac{V}{t_0} = \frac{(2n-1)mL}{(n+Lm-1)2\alpha}$$

Данные на выходе систолического процессора появляются через такт работы схемы. Соответственно производительность его будет в два раза ниже по сравнению с первым вариантом.

5.1.2. Матричные систолические процессоры

Для повышения производительности вычислений можно объединить несколько линейных процессоров в двухмерную структуру. Так, рассмотренный на рис.5.4 систолический процессор реализует функцию одного нейронного элемента. При объединении таких процессоров в соответствии с количеством m нейронных элементов получается систолический массив размерностью nxm. Такой процессор размерностью n = m =3 представлен на рис. 5.11.

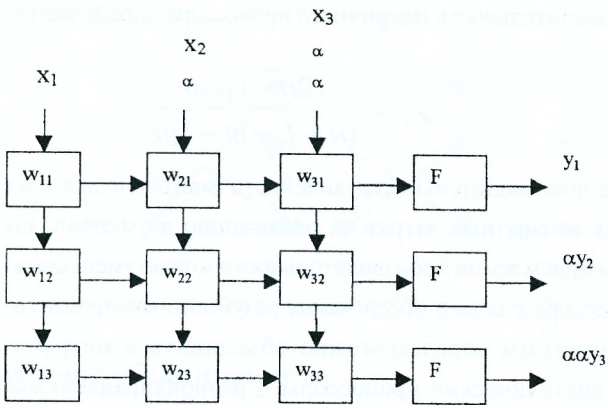


Рис.5.11. Матричный процессор

Весовые коэффициенты нейронной сети распределены по соответствующим процессорным элементам, как показано на рисунке. Схема процессорного элемента для матричного массива представлена на рис.5.12.

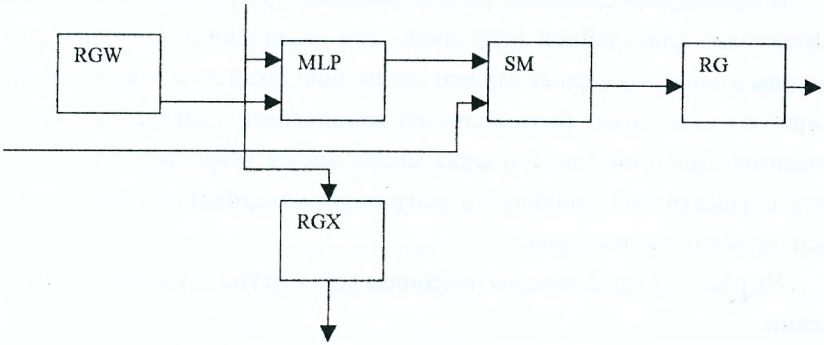


Рис.5.12. Схема процессорного элемента: RGW – регистр для хранения весовых коэффициентов; RGX – регистр для хранения входных данных

Производительность матричного процессора определяется как

$$\gamma_m = \frac{(2n-1)Lm}{(n+L+m-1)\alpha},$$

что больше производительности линейного систолического массива. Для уменьшения аппаратных затрат на реализацию двумерного процессора и достижения приемлемой производительности можно уменьшить количество строк процессора с целью обеспечения разумного компромисса.

Аналогичным образом можно объединить в матричный массив линейные систолические процессоры с разнонаправленными связями. Рассмотренные процессоры характеризуются простой реализацией и легко отображаются на СБИС-технологию.

5.2. Многослойные нейронные сети

В предыдущих разделах рассматривались три различных варианта отображения однослойной нейронной сети на линейные систолические массивы и один двухмерный вариант реализации такой сети. Многослойная нейронная сеть может быть получена компиляцией приведенных выше вариантов. При этом для получения многослойной нейронной сети может быть осуществлено линейное и матричное объединение одномерных систолических процессоров.

На рис. 5.13 представлена нейронная сеть с двумя обрабатывающими слоями.

Рассмотрим отображение такой сети на систолические массивы.

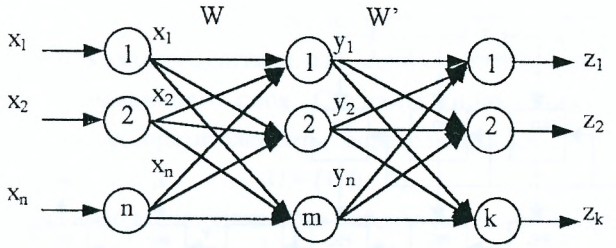


Рис. 5.13. Нейронная сеть с двумя обрабатывающими слоями

5.2.1. Линейное объединение

Результатом такого объединения систолических процессоров является также одномерный массив.

Существует два варианта линейного объединения: непосредственный и с использованием обратной связи.

Непосредственное объединение характеризуется наращиванием одномерных процессоров в соответствии с количеством слоев нейронной сети. При этом объединяться могут как идентичные, так и различные систолические процессоры. Пример реализации двухслойной нейронной сети при объединении различных процессоров приведен на рис.5.14.

Коэффициенты W и W' характеризуют соответственно синаптические связи первого и второго слоя нейронной сети. Весовые коэффициенты W' начинают подаваться на процессорные элементы PE', как только будут сформированы выходные значения первой линейки. Недостатком такой схемы является разная производительность линеек процессоров. Согласно закону Амдала, в системе с быстрым и медленным процессором низкоскоростной режим доминирует в общей производительности системы. Поэтому второй процессор будет снижать производительность системы,

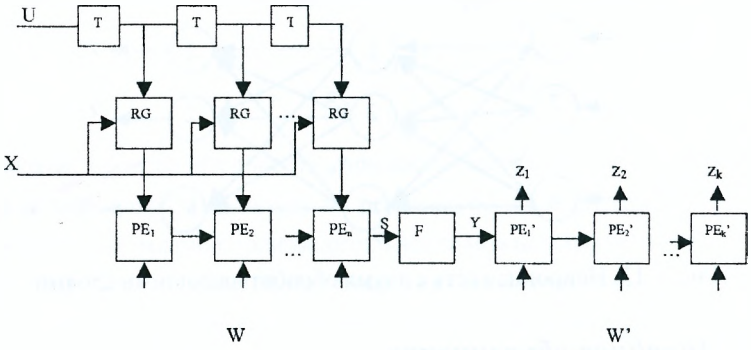


Рис.5.14. Реализация двухслойной нейронной сети при объединении различных систолических процессоров

особенно при смене входных данных X.

На рис.5.15 приведен пример реализации двухслойной нейронной сети при объединении идентичных процессоров.

Количество процессорных элементов в таком массиве равняется:

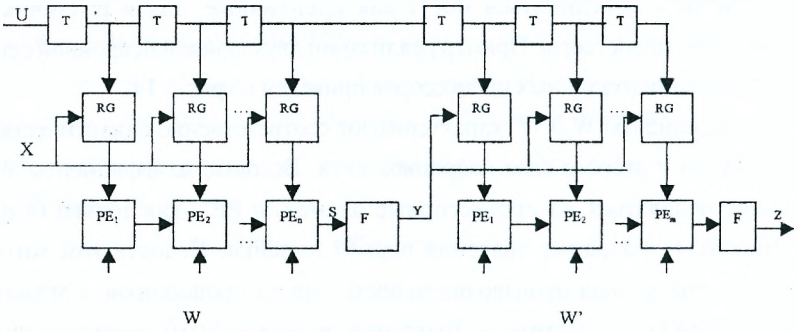


Рис.5.15. Реализация двухслойной нейронной сети при объединении идентичных систолических процессоров

$$N = n + m,$$

где n - размерность входного сигнала, а m - количество нейронных элементов первого обрабатывающего слоя. Сигналы управления занесением информации в процессорные элементы определяются как

$$U = (100...0)$$

Данные на выходе систолического массива появляются в каждом такте работы схемы. Поэтому такой вариант объединения имеет преимущество по производительности по сравнению с предыдущим вариантом.

Линейное объединение с использованием обратной связи характеризуется незначительными аппаратными затратами. В этом случае одни и те же процессорные элементы используются для реализации функций различных слоев нейронной сети. Так как размерность слоев нейронной сети может быть различной, то необходимо иметь возможность варьировать количеством функционирующих процессорных элементов в линейке. На рис.5.16 показан фрагмент систолической схемы для реализации многослойной нейронной сети.

Для такой схемы сигналы управления цепочкой триггеров формируются следующим образом:

$$U=\{u_1, u_2 \dots u_c\},$$

где $u_1=u_2= \dots =u_c$; $u_1=(100)$.

Количество таких сигналов определяется как

$$c = p - 1,$$

где p - количество слоев нейронной сети.

Для регулирования размерности слоев нейронной сети в схему введены ключи SW с тремя состояниями, которые управляются триггерами Tr.

Если триггер Tr находится в нулевом состоянии, то ключ SW отключен и соответствующий ему процессорный элемент работает в обычном режиме. При переходе триггера в единичное состояние, отключается

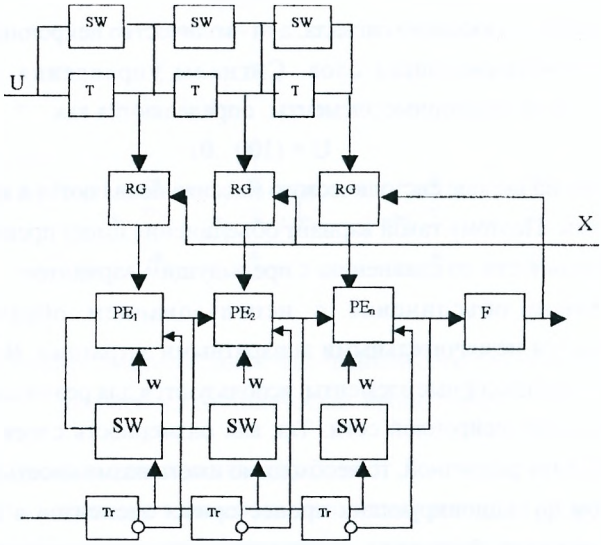


Рис.5.16. Фрагмент систолической схемы для реализации многослойной нейронной сети

соответствующий процессорный элемент из систолического массива. Таким образом при помощи занесения соответствующей информации в цепочку триггеров Tr можно регулировать размерность систолического процессора и реализовать функции различных слоев нейронной сети. В схеме, показанной на рис.5.16 предполагается также, что процессорные элементы на выходе имеют регистры с тремя состояниями, которые управляются инверсными выходами триггеров Tr. Необходимо также при реализации различных слоев нейронной сети соответствующим образом изменять матрицу весовых коэффициентов.

На рисунке 5.17 представлена другая схема организации систолического массива с обратной связью.

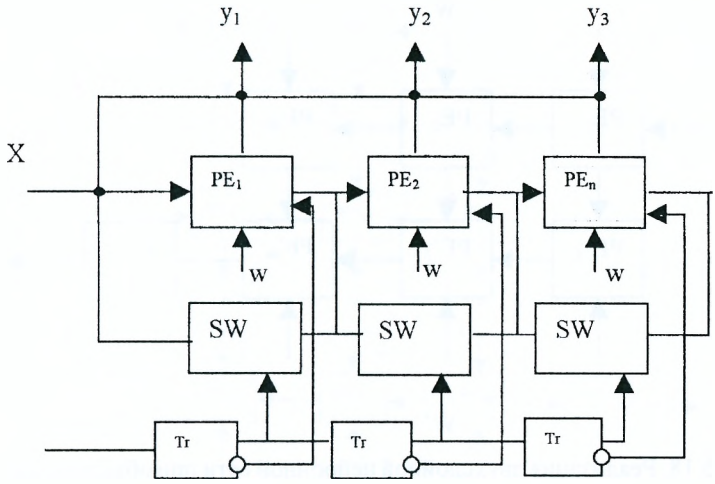


Рис.5.17. Систолический процессор с обратными связями для реализации многослойной нейронной сети

Здесь предполагается, что обратная связь начинает действовать при появлении сигналов $Y=(y_1, y_2 ...)$ на выходах процессорных элементов. Это осуществляется путем соответствующего управления регистрами процессоров. Аналогичным образом можно реализовать многослойную нейронную сеть при помощи систолического процессора с разнонаправленными связями.

5.2.2. Матричное объединение

Характеризуется тем, что в результате объединения линейных процессоров получается двухмерный систолический массив. При этом одномерные массивы играют роль атомарных элементов, из которых строится матричный систолический процессор. На рис.5.18 изображен матричный

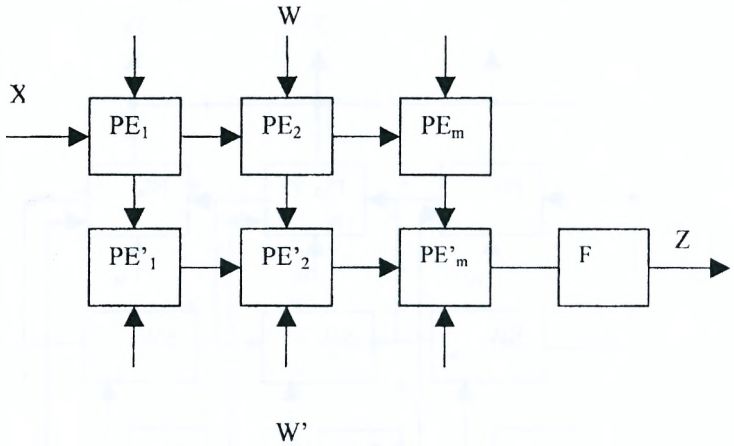


Рис.5.18. Реализация двухслойной нейронной сети при объединении линейных процессоров; W и W' – соответственно матрицы весовых коэффициентов для первого и второго обрабатывающих слоев нейронной сети

процессор для реализации двухслойной нейронной сети. Он получается путем объединения в матрицу двух различных линейных массивов (рис. 5.7 и 5.4) и содержит 2m процессорных элементов.

Процессорные элементы PE_i выполняют роль первого слоя, а процессорные элементы PE_i'-второго слоя нейронной сети.

Другим вариантом построения многослойных нейронных сетей является объединение между собой матричных и линейных систолических массивов. На рис. 5.19 изображен двухмерный систолический массив, который получается путем объединения матричного процессора (рис. 5.11) и линейного процессора (рис. 5.4).

Такая систолическая структура реализует функции двухслойной нейронной сети. Здесь предполагается, что весовые коэффициенты первого

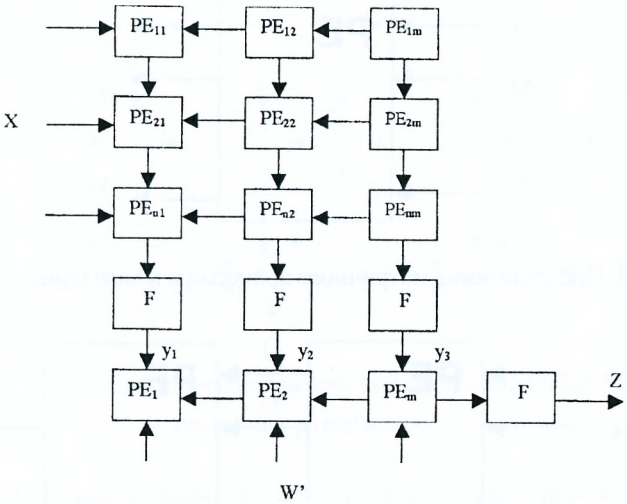


Рис.5.19. Объединение матричного и линейного процессора

слоя X хранятся в регистрах соответствующих процессорных элементов PE_{ij}. Размерность систолического массива равняется m(n+1).

Двухмерные систолические структуры для реализации многослойных нейронных сетей можно получать также при объединении матричных процессоров. Так, если систолический процессор, изображенный на рис.5.11 представить в виде одного элемента (рис.5.20), то путем объединения таких элементов можно реализовать нейронную сеть с произвольным количеством слоев.

Пример реализации двухслойной нейронной сети изображен на рис.5.21.

Размерность такого процессора равняется m(n+k). В данном разделе рассмотрены различные схемы реализации нейронных сетей на систолических массивах. Они отличаются между собой сложностью и быстродействием. Выбор того или иного варианта зависит от технических

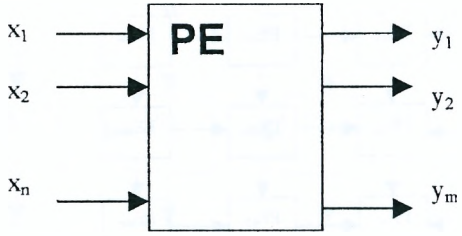


Рис.5.20. Представление матричного процессора в виде одного элемента

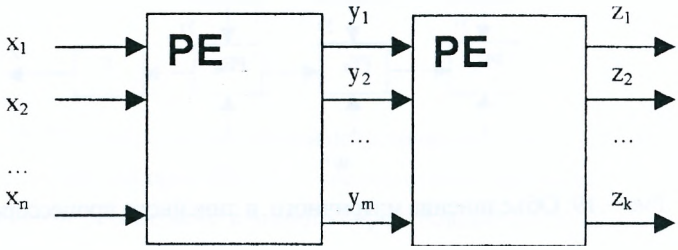


Рис.5.21. Матричный систолический процессор для реализации двухслойной нейронной сети

требований, предъявляемых к систолическому массиву.

5.3. Релаксационные нейронные сети

В предыдущих разделах рассматривалась технология отображения нейронных сетей с прямыми связями на систолические массивы. Аналогичным образом, с учетом специфики нейронной сети, происходит реализация различных типов сетей на систолических процессорах. В релаксационных нейронных сетях обработка информации происходит до тех пор, пока не перестанут изменяться состояния нейронных элементов.

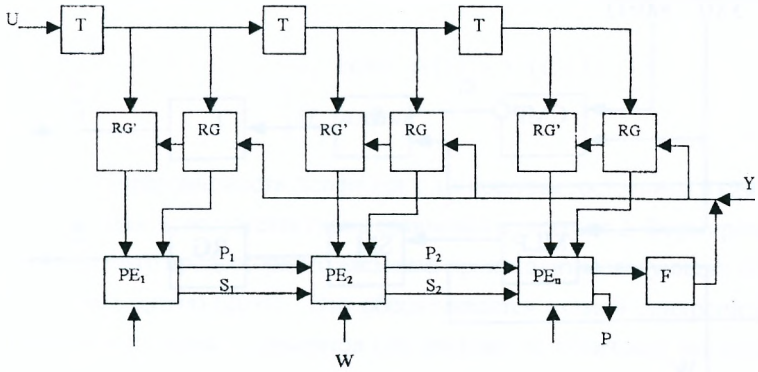


Рис.5.22. Систолическая реализация нейронной сети Хопфилда:P–фиксирует состояние равновесия

Рассмотрим отображение нейронной сети Хопфилда на систолический массив. Каждый нейронный элемент такой сети изменяет свое состояние в соответствии со следующим выражением:

$$y_i(t+1) = F(\sum_{j}^{n} w_{ji} y_j(t)).$$

где F - оператор нелинейного преобразования, n - количество нейронных элементов. Обработка информации заканчивается когда $y_i(t+1)=y_i(t)$ для всех нейронных элементов. В этом случае сеть переходит в стабильное состояние. Исходя из этого необходимо фиксировать в систолическом массиве состояние равновесия. На рис.5.22 изображен линейный систолический процессор, который реализует нейронную сеть Хопфилда.

Занесение информации в регистры RG осуществляется под управлением цепочки триггеров T. В первые n тактов функционирования процессора происходит запись в регистры RG компонентов неизвестного образа

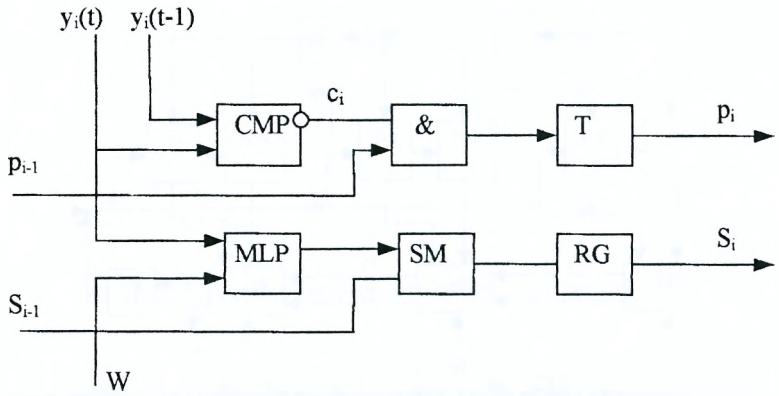


Рис.5.23. Схема процессорного элемента: CMP– схема сравнения

$Y(0)=(y_1(0),y_2(0),\ldots,y_n(0))$, который поступает с внешнего устройства. Параллельно производится обработка соответствующей информации процессорными элементами. В дальнейшем данные в регистры RG поступают с выхода нелинейного преобразователя F при помощи обратной связи. Так, при истечении n+1 тактов, на выходе нелинейного преобразователя F будет сформирована компонента $y_1(1)$, которая через обратную связь записывается в регистр RG первого процессорного элемента. Одновременно предыдущее значение $y_1(0)$ записывается в регистр RG' соответствующего процессорного элемента. Процессорный элемент сравнивает значения $y_1(0)$ и $y_1(1)$ для формирования сигнала равновесия P. Сигнал P=1 на выходе процессорного элемента PE_n фиксирует стабильное состояние нейронной сети и прерывает процесс вычислений. Выходные данные при этом, характеризующие распознанный образ будут находиться в регистрах RG. Схема процессорного элемента представлена на рис.5.23.

Схема сравнения реализует следующую функцию

$$C_i = \begin{cases} 1, & \textit{если } y_i(t) = y_i(t-1) \\ 0, & \textit{иначе} \end{cases}$$

В случае нейронной сети Хопфилда с дискретным состоянием в качестве схемы сравнения используется схема сравнения по модулю 2. Формирование сигнала p_i для i-го процессорного элемента происходит в следующих тактах работы схемы: i,(n+i),(2n+i)…Это обеспечивается схемой синхронизации триггеров процессорных элементов (на рисунке не показана), на которые синхросигналы поступают с выходов соответствующих триггеров сдвигающего регистра (рис. 5.22).

Сигнал $P=f(p_1,p_2,...,p_n)$ формируется тогда, когда все процессорные элементы перестают изменять свое состояние. Рассмотренная схема систолического процессора характеризуется двухрядными связями между процессорными элементами. Она позволяет легко реализовать сеть Хопфилда на микроэлектронной технологии. Путем объединения таких линейных массивов можно получить систолические массивы для других релаксационных сетей.

ГЛАВА 6. САМООРГАНИЗАЦИЯ И ОТКАЗОУСТОЙЧИВОСТЬ

В главе 1 тома 1отмечалось, что под самоорганизацией понимается способность организма перестраиваться с целью адаптации к внешней среде. При этом происходит изменение структуры организма (меняются синаптические связи головного мозга) и его связей с внешней средой.

Самоорганизация нейронных систем характеризуется как способностью адаптации синаптических связей к решаемой задаче, так и возможностью порождения сетью топологически упорядоченных структур в результате взаимодействия с внешней средой (сети Кохонена).

В данной главе рассматривается структурная самоорганизация нейронных сетей, реализованных на линейных систолических процессорах. Структурной самоорганизацией называется динамическая перестройка структуры схемы с целью адаптации к различным условиям функционирования. Она может происходить как для нейтрализации отказавших элементов, так и для адаптации схемы к размерности решаемой задачи. Способность схемы к нейтрализации отказавших элементов для сохранения работоспособности характеризуется ***отказоустойчивостью***. Таким образом отказоустойчивость является частным случаем самоорганизации, которая происходит с целью обеспечения работоспособности схемы. Аналогичные процессы наблюдаются в живых организмах. Как отмечалось в разделе 1.4 тома 1, при утрате с возрастом нейронов головного мозга происходит перестройка синаптических связей, в результате которой жизнеспособные нейроны стремятся компенсировать эту утрату.

Другим аспектом этой проблемы является то, что самоорганизация,

ориентированная на отказоустойчивость, является эффективным средством как для повышения выхода годных, так и улучшения надежностных характеристик интегральных схем [36]. Это позволяет проектировать схемы с интеграцией на уровне пластины (wafer scale integration), где выход годных является критическим фактором. Для определения способа обеспечения структурной самоорганизации необходимы количественные оценки показателей отказоустойчивости схемы. При проектировании схемы на уровне кристалла такими оценками являются выход годных и показатели надежности.

6.1 Статистические модели выхода годных

Под выходом годных СБИС понимается отношение числа бездефектных кристаллов, к общему числу кристаллов на пластине. Большинство производственных отказов приходится на точечные дефекты, которые являются локализованными и случайным образом распределенными на пластине [44]. Имеется единое мнение, что дефекты имеют тенденцию к группированию. Для описания распределения дефектов в пределах кластера обычно используется пуассоновское распределение [45]. Тогда вероятность возникновения в кластере x дефектов:

$$Y_k(x) = \frac{e^{-\lambda_K}\lambda^x}{x!}, \qquad (6.1)$$

где λ_k–среднее число неисправностей на кристалле в кластере.

В случае бесконечного числа кластеров вероятность возникновения на кристалле x дефектов равняется:

$$p(x) = \int_0^\infty Y_k(x) f(D) dD = \int_0^\infty \exp(-\lambda_k) f(D) dD,$$

где f(D)–функция плотности вероятности среднего числа неисправностей на кристалле в каждом кластере. Если в качестве f(D) использовать гамма распределение, то:

$$p(x) = \frac{\Gamma(x+\alpha)(\lambda/\alpha)^x}{x!\,\Gamma(\alpha)(1+\lambda/\alpha)^{x+\alpha}}, \qquad (6.2)$$

где λ=AD – среднее число неисправностей на кристалле, A – площадь кристалла; D – средняя плотность дефектов на кристалле; α – параметр группирования дефектов; Γ – гамма функция.

Выражение (6.2) называется обобщенным отрицательным биномиальным распределением. Выход годных можно определить как вероятность отсутствия дефектов на кристалле. Тогда

$$Y = p(x=0) = \frac{1}{(1+\lambda/\alpha)^\alpha}. \qquad (6.3)$$

Значения λ и α зависят от параметров технологической линии и определяется по методу "окна" при помощи нелинейного регрессионного анализа [45].

Рассмотрим определение выхода годных для отказоустойчивых схем [46-48]. Не касаясь конкретного метода обеспечения отказоустойчивости, предположим , что путем структурной перестройки можно нейтрализовать в схеме R неисправных элементов без потери работоспособности. Предположим, что в элемент схемы может попасть бесконечное число дефектов. Тогда выражение для определения выхода годных можно

представить в следующем виде:

$$Y(R) = \sum_{x=0}^{\infty} \sum_{i=0}^{R} k(i) p_X(i) p(x), \qquad (6.4)$$

где $p_x(i)$ – вероятность попадания дефектов в фиксированные i элементов схемы; p(x) – вероятность возникновения на кристалле X производственных дефектов; k(i) – число разрешенных сочетаний i элементов из общего количества элементов схемы.

В качестве p(x) в выражении (6.4) используется обобщенное отрицательное биномиальное распределение. ***Разрешенными сочетаниями*** назовем такие сочетания элементов, выбор которых не приводит к отказу схемы. Из определения следует, что число разрешенных сочетаний элементов определяется методом обеспечения отказоустойчивости и соответственно способом организации структурной перестройки схемы.

Пусть $P_{x,N_r}(i)$ – вероятность попадания x производственных дефектов в i элементов из N, где N_r=N+R – общее количество элементов на кристалле; R – количество элементов, которые могут отказать без потери работоспособности схемы (резервные элементы); N – число элементов исходной, без средств отказоустойчивости, схемы. Тогда

$$P_{x,N_r}(i) = k(i) p_x(i). \qquad (6.5)$$

При кластеризации дефектов на большой площади (размеры кластера больше размеров кристалла) в качестве $p_x(i)$ используется выражение, базирующееся на статистике Максвелла-Больцмана [47]. В этом случае

$$p_x(i) = \sum_{r=0}^{i} (-1)^r \binom{i}{r} \left(\frac{i-r}{N+R}\right)^x . \qquad (6.6)$$

Статистика Максвелла-Больцмана исходит из предположения различимости дефектов на кристалле.

При кластеризации дефектов на малой площади в качестве $p_x(i)$ используется статистика Бозе-Энштейна, которая базируется на предположении неразличимости дефектов на кристалле [47]:

$$p_x(i) = \binom{x-1}{i-1} : \binom{x+N+R-1}{x} . \qquad (6.7)$$

Определим число разрешенных сочетаний элементов схемы для различных методов обеспечения отказоустойчивости.

Пусть схема разбита на секции и в каждой секции независимо друг от друга может осуществляться структурная перестройка с целью нейтрализации отказавших элементов секции. Обозначим число секций через N_c, а число элементов в секции – N_o. Предположим, что в каждой секции существует $R_c=R/N_c$ резервных элементов, которые в случае необходимости могут заменить любой из элементов данной секции. Назовем такой метод структурной перестройки схемы секционированным.

Теорема 6.1. Для секционированного резервирования число разрешенных сочетаний i элементов из общего количества элементов схемы определяется на основе следующего выражения:

$$k(i) = \sum_{J_1, J_2 \ldots J_r} \prod_{l=0}^{r-1} \binom{J_l}{J_{l+1}} \left(\frac{N_o + r - l}{l+1}\right)^{J_{l+1}} , \qquad (6.8)$$

где $J_o=N_c$, $r=R_c$, $i=\sum_{l=1}^{r} J_l$. Значения J_l, $l=\overline{1,r}$ определяются при этом путем раскрытия выражения (11.9) и определения J_l, $l=\overline{1,r}$ при равных значениях i:

$$b=\sum_{J_1=0}^{J_o}\sum_{J_2=0}^{J_1}\ldots\sum_{J_r=0}^{J_{r-1}}(J_1+J_2+\ldots J_r)\cdot \qquad (6.9)$$

Доказательство. Пусть y_1 – выход годных одного элемента на кристалле. Тогда при секционированном резервировании выход годных всей схемы можно определить следующим образом:

$$Y_R=\left(y_1^{N_O+r}+(N_o+r)y_1^{N_O+r-1}(1-y_1)+\binom{N_o+r}{2}y_1^{N_O+r-2}(1-y_1)^2+\ldots\binom{N_o+r}{r}y_1^{N_O}(1-y_1)^r\right)^{N_C}$$

Раскладывая последовательно данное выражение по формуле Ньютона, получим

$$Y_R=\sum_{J_1=0}^{N}\binom{N_c}{J_1}(N_o+r)^{J_1}y_1^{N+R-J_1}(1-y_1)^{J_1}\sum_{J_2=0}^{J_1}\binom{J_1}{J_2}\times$$

$$\times\left(\frac{N_o+r-1}{2}\right)^{J_2}\left(\frac{1-y_1}{y_1}\right)^{J_2}\ldots\sum_{J_r=0}^{J_{r-1}}\left(\frac{N_o+1}{r}\right)^{J_r}\binom{J_{r-1}}{J_r}\left(\frac{1-y_1}{y_1}\right)^{J_r}=$$

$$=\sum_{J_1=0}^{N_C}\sum_{J_2=0}^{J_1}\ldots\sum_{J_r=0}^{J_{r-1}}\binom{N_c}{J_1}(N_o+r)^{J_1}y_1^{N+R-J_1}\times$$

$$\times (1-y_1)^{J_1} \prod_{l=1}^{r-1} \binom{J_l}{J_{l+1}} \left(\frac{N_o + r - l}{l+1} \right)^{J_{l+1}} \left(\frac{1-y_1}{y_1} \right)^{J_{l+1}}, \qquad (6.10)$$

где N – число элементов исходной, без средств отказоустойчивости схемы.

Выражение для определения выхода годных схем при использовании биномиального распределения можно представить также в следующем виде:

$$Y_R = \sum_{i=0}^{R} k(i) y_1^{N+R-i} (1-y_1)^i \cdot \qquad (6.11)$$

Из анализа выражения (6.10) и (6.11) видно, что количество разрешенных сочетаний i элементов из N+R будет равняться сумме всех коэффициентов при одинаковых степенях y_1 и $(1-y_1)$. Поэтому, производя операцию выделения из выражения (6.10) соответствующих коэффициентов, получим искомую теорему.

Согласно теореме 6.1 алгоритм вычисления числа разрешенных сочетаний элементов схемы можно свести к следующей последовательности шагов:

1) раскрывается выражение (6.9);

2) определяются значения $J_1, J_2 \ldots J_r$ при равных значениях i;

3) на основе выражения (6.8) вычисляется число различных сочетаний элементов схемы $k(i)$ при отказе i элементов.

Пример 11.1. Пусть дана схема, изображенная на рис. 11.1. Здесь $J_o = 3$, $r = 2$, $i = J_1 + J_2$. Определение числа разрешенных сочетаний, согласно теореме 11.1, можно свести к следующей последовательности шагов.

1. Раскрываем выражение (6.9):

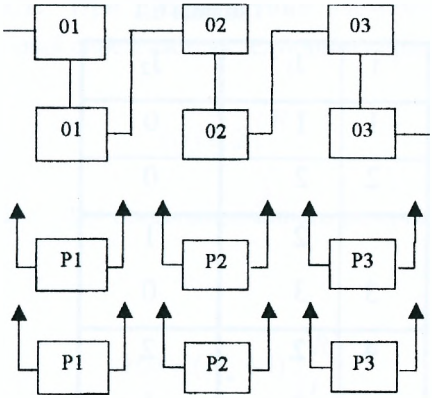


Рис. 6.1 Структурная схема секционированного резервирования: 01, 02, 03 – соответственно основные элементы 1-ой, 2-ой и 3-ей секции; P1, P2, P3 – резервные элементы соответствующих секций

$$b = \sum_{J_1=0}^{3}\sum_{J_2=0}^{J_1}(J_1 + J_2) = (0+0)+(1+0)+(1+1)+(2+0)+(2+1)+(2+2)+(3+0)+$$

$$+(3+1)+(3+2)+(3+3).$$

2. Определим значения J_1, J_2 при равных значениях i. Представим их результаты в таблице 6.1:

3. На основе выражения (6.8) определяем число разрешенных сочетаний элементов схемы k(i) при отказе любых i элементов.

Пусть, например, i = 3. Тогда

$$k(3) = \binom{3}{2}4^2\binom{2}{1}\frac{3}{2} + \binom{3}{3}4^3\binom{3}{0}\left(\frac{3}{2}\right)^0 = 208 \cdot$$

Секционированный метод обеспечения отказоустойчивости

Таблица 6.1

i	J_1	J_2
1	1	0
2	2	0
3	2 3	1 0
4	2 3	2 1
5	3	2
6	3	3

характеризует локальный подход к структурной перестройке схемы, когда не все элементы являются структурно взаимозаменяемыми. Степень локальности определяется количеством элементов в секции N_o, каждый из которых в случае необходимости может быть заменен любым резервным элементом данной секции. Тогда N_o характеризует метод обеспечения отказоустойчивости. Так, если $N_o = 1$ и $Rc = 1$, то для элементов схемы используется дублирование. При $N_o \neq 1$ и $Rc \neq N$ для элементов схемы используется различного рода секционированное резервирование, а при $N_o = N$ – скользящее резервирование. Скользящее резервирование характеризуется глобальным подходом к структурной перестройке схемы, когда все элементы являются взаимозаменяемыми. Используя результаты приведенной выше теоремы можно определить число разрешенных сочетаний для различных методов обеспечения отказоустойчивости.

Следствие 6.1. При дублировании элементов схемы число разрешенных сочетаний определяется следующим образом:

$$k(i) = 2^i \binom{R}{i}. \qquad (6.12)$$

Следствие 6.2. Для секционированного резервирования, в случае когда $Rc = 1$:

$$k(i) = (N_o + 1)^i \binom{R}{i}. \qquad (6.13)$$

Следствие 6.3. При скользящем резервировании число разрешенных сочетаний схемы определяется как:

$$k(i) = \binom{N+R}{i}. \qquad (6.14)$$

Определяя число разрешенных сочетаний и используя выражение (6.4) можно определить выход годных схем для различных методов обеспечения отказоустойчивости.

В случае использования в выражении (6.4) в качестве $p_x(i)$ статистики Максвелла-Больцмана его можно представить следующим образом [113].

$$Y(R) = \sum_{i=0}^{R} k(i) \sum_{r=0}^{i} (-1)^r \binom{i}{r} \left(1 + \frac{(N+R-i+r)A_1 D}{\alpha}\right)^{-\alpha}, \qquad (6.15)$$

где A_1 – площадь одного элемента на кристалле. В случае линейных систолических процессоров деление элементов схемы на резервные и основные является довольно условным, так как все они могут участвовать в

процессе вычислений. При отказе элементов схема деградирует в сторону уменьшения производительности или уменьшения ее способности адаптироваться к задачам различной размерности. Приведенные выше выражения позволяют оценивать выход годных для различных методов структурной перестройки схемы.

С целью эффективного повышения выхода годных схемы необходимо обеспечить увеличение съема кристаллов с пластины. Для количественной оценки съема кристаллов с пластины можно использовать показатель качества резервирования:

$$F(R) = \frac{Y(R)A(0)}{Y(0)A(0)},$$

где Y(R), Y(0) – соответственно выход годных для отказоустойчивой и исходной схемы; A(R), A(0) – площадь кристалла для отказоустойчивой и исходной схемы.

Для увеличения съема кристаллов с пластины необходимо, чтобы показатель качества резервирования был больше единицы.

6.2. Показатели надежности

Как уже отмечалось, отказоустойчивость может быть ориентирована как на нейтрализацию производственных отказов с целью повышения выхода годных схем, так и на нейтрализацию эксплуатационных отказов для улучшения показателей надежности. Рассмотрим расчет показателей надежности для невосстанавливаемых схем, когда все элементы находятся в рабочем состоянии.

Пусть имеется гипотетическая схема, в которой может отказать R

элементов без потери ее работоспособности. Тогда вероятность безотказной работы такой схемы можно определить следующим образом:

$$P(R) = \sum_{i=\partial}^{R} k(i) p_1^{N+R-i} (1 - p_1)^i , \qquad (6.16)$$

где p_1 – вероятность безотказной работы одного элемента схемы.

При использовании экспоненциального закона надежности:

$$p_1 = \exp(-\lambda_1 t), \qquad (6.17)$$

где l_1 –интенсивность отказов одного элемента схемы.

Средняя наработка схемы до отказа определяется как:

$$T(R) = \int_{\partial}^{\infty} P(R)dt = \frac{1}{\lambda_1} \sum_{i=0}^{R} \frac{k(i)}{(N + R - i)\binom{N+R}{i}}. \qquad (6.18)$$

Из выражений (6.16) и (6.18) следует, что показатели надежности схемы при прочих равных условиях определяются количеством разрешенных сочетаний. Выражения (6.16) и (6.18) являются инвариантными относительно метода обеспечения отказоустойчивости. Определяя на основе теоремы 6.1 значение k(i) для соответствующего метода обеспечения отказоустойчивости и подставляя его в приведенные выше выражения, можно определить показатели надежности схемы. Используя количественные показатели, рассмотренные в этом и предыдущем разделе, можно оценить различные способы организации структурной перестройки схемы и выбрать среди них оптимальный.

6.3. Факторы, влияющие на структурную реконфигурацию линейных систолических процессоров

Самоорганизация систолических массивов происходит посредством реконфигурации структуры схемы, в результате чего обеспечивается отказоустойчивость. При этом необходимо при приемлемых аппаратных издержках обеспечить нейтрализацию отказавших элементов. Способы структурной организации обхода неисправных элементов в процессорном массиве и соответственно аппаратные затраты на обеспечение отказоустойчивости зависят от организации потоков входных данных [43,49,50].

В линейных систолических структурах может существовать несколько потоков входных данных, организованных различным образом. Соответственно для различных потоков входных данных возможны различные методы обхода неисправных элементов. В систолических структурах можно выделить следующие типы организации потока входных данных: последовательный поток, параллельный однородный и параллельный неоднородный поток входных данных. При этом различные типы данных, подаваемые в систолический массив могут быть организованы различными способами в соответствии с приведенной выше классификацией. Организацию входного потока данных можно представить в виде матрицы $M = [m_{ij}]$ (i = ; j =), характеризующей распределение этого потока по времени и процессорным элементам. Элемент, стоящий в i-й строке и j-м столбце такой матрицы характеризует, что в момент времени t_i на процессорный элемент Pi,

подается соответствующий элемент данных. Нулевые элементы матрицы свидетельствуют об отсутствии подачи входных данных на соответствующие процессорные элементы, число столбцов матрицы характеризует общее количество процессорных элементов в линейке, а количество различных элементов матрицы (за исключением нулевых) - размерность входного потока данных. Различные структуры линейных систолических процессоров характеризуются разными матрицами потока входных данных. Так организация систолической структуры с последовательным потоком входных данных характеризуется тем, что входные данные последовательно вводятся через граничный процессорный элемент, а затем они (или результаты их обработки) последовательно передаются от одного процессорного элемента к другому. В этом случае первый столбец матрицы распределения потока входных данных содержит полностью все элементы входного потока и ненулевые элементы любой из строк матрицы различны между собой.

К систолическим структурам, у которых имеется такая организация входного потока данных, относятся процессоры для реализации однослойной нейронной сети (рис. 5.7, рис. 5.9) и многослойной сети (рис. 5.17). Так для систолического процессора, изображенного на рис. 5.7 фрагмент матрицы распределения для потока входных данных X будет иметь следующий вид:

$$X = \begin{array}{c} \\ t_1 \\ t_2 \\ t_3 \\ t_4 \end{array} \begin{array}{c} \begin{array}{cccc} P_1 & P_2 & P_3 & P_4 \end{array} \\ \begin{bmatrix} x_1 & 0 & 0 & 0 \\ 0 & x_1 & 0 & 0 \\ x_2 & 0 & x_1 & 0 \\ 0 & x_2 & 0 & x_1 \end{bmatrix} \end{array}$$

Организация систолической структуры с параллельным

однородным потоком входных данных характеризуется тем, что пространство ввода содержит все процессорные элементы, а столбцы или строки матрицы распределения содержат одинаковые ненулевые элементы.

Так для систолического процессора, представленного на рис. 5.4. фрагмент матрицы распределения для потока входных X:

$$X = \begin{array}{c} \\ t_1 \\ t_2 \\ t_3 \\ t_4 \end{array} \begin{array}{c} \begin{array}{cccc} P_1 & P_2 & P_3 & P_4 \end{array} \\ \begin{bmatrix} x_1 & 0 & 0 & 0 \\ x_1 & x_2 & 0 & 0 \\ x_1 & x_2 & x_3 & 0 \\ x_1 & x_2 & x_3 & x_4 \end{bmatrix} \end{array}.$$

В качестве примера структур, у которых имеется параллельный однородный поток входных данных, можно привести систолический процессор для реализации нейронной сети Хопфилда (рис. 5.22).

Организация систолической структуры с параллельным неоднородным потоком входных данных характеризуется также пространством ввода, содержащим все процессорные элементы, но при этом ненулевые элементы столбцов и строк матрицы распределения различны между собой. К структурам у которых имеется параллельный неоднородный поток входных данных относятся систолические процессоры, изображенные на рисунках 5.4, 5.7 и 5.22. Так фрагмент матрицы распределения потока входных данных W для процессора представленного на рис. 5.4 имеет следующий вид.

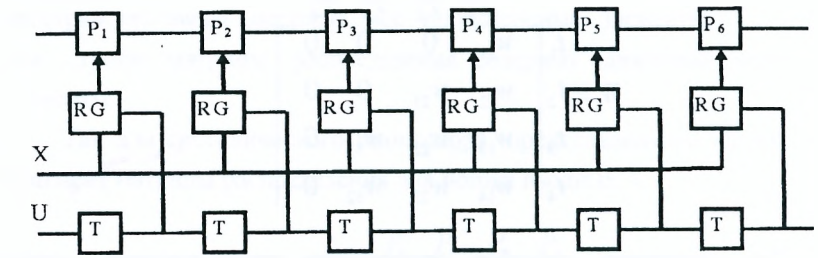
$$W = \begin{array}{c} \\ t_1 \\ t_2 \\ t_3 \\ t_4 \end{array} \begin{array}{c} \begin{array}{cccc} P_1 & P_2 & P_3 & P_4 \end{array} \\ \begin{bmatrix} w_{11} & 0 & 0 & 0 \\ w_{12} & w_{21} & 0 & 0 \\ w_{13} & w_{22} & w_{31} & 0 \\ w_{14} & w_{23} & w_{32} & 0 \end{bmatrix} \end{array}$$

Для различных способов организации потока входных данных существуют различные методы обхода неисправных процессорных элементов. Рассмотрим их для разных типов организации потока входных данных.

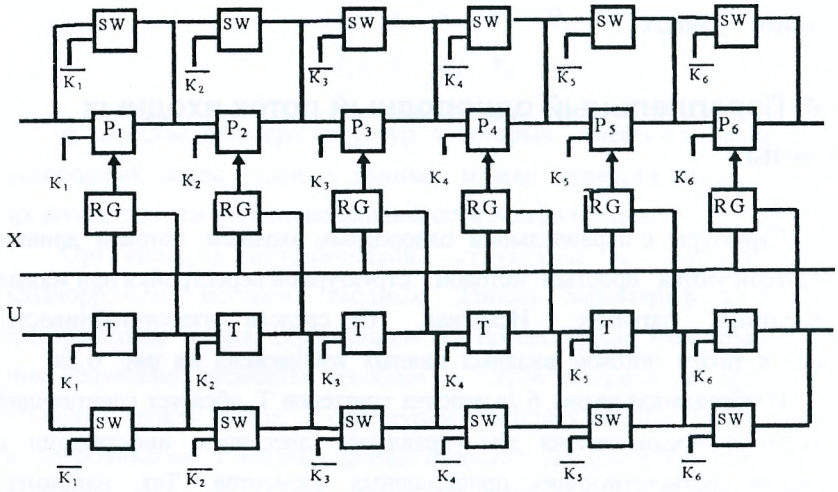
6.4. Параллельный однородный поток входных данных

Структуры с параллельным однородным входным потоком данных характеризуются простым методом структурной перестройки при малых аппаратных затратах. Исходная, без средств отказоустойчивости, схема с таким потоком входных данных изображена на рис. 6.2а.

Изображенная на рис. 6.2а цепочка триггеров Т, образует сдвигающий регистр и предназначена для управления занесением информации в регистры соответствующих процессорных элементов. Так, например, если поток входных данных $X = (X_1, X_2, ..., X_6)$, а сигналы управления $U = (1, 0, ..., 0)$, то в первом такте t_1, в регистр RG процессорного элемента P_1, занесется информация X_1, во втором такте t_2 в регистр RG процессорного элемента P_2, - X_2, и т.д. Таким образом, фрагмент матрицы распределения потока входных данных можно представить в следующем виде:



а



б

Рис. 6.2. Исходная схема (**а**) и отказоустойчивая (**б**) схема с параллельным однородным потоком входных данных: **X** - входной поток данных; **U** - сигналы управления; **Pi** - i-ый процессорный элемент.

$$X = \begin{array}{c} \\ t_1 \\ t_2 \\ t_3 \\ t_4 \\ t_5 \end{array} \begin{array}{c} \begin{array}{cccccc} P_1 & P_2 & P_3 & P_4 & P_5 & P_6 \end{array} \\ \begin{bmatrix} X_1 & 0 & 0 & 0 & 0 & 0 \\ X_1 & X_2 & 0 & 0 & 0 & 0 \\ X_1 & X_2 & X_3 & 0 & 0 & 0 \\ X_1 & X_2 & X_3 & X_4 & 0 & 0 \\ X_1 & X_2 & X_3 & X_4 & X_5 & 0 \end{bmatrix} \end{array}$$

Рассмотрим обеспечение отказоустойчивости такой схемы при использовании скользящего резервирования с алгоритмом реконфигурации "сдвиг в линейке". Для этого в исходную схему вводятся ключи SW с тремя состояниями для обхода триггеров Т сдвигающего регистра с целью перераспределения входного потока данных X на исправные процессорные элементы, а также для обхода неисправных процессорных элементов с целью перераспределения промежуточных данных между процессорными элементами (рис. 6.2б). Сигналы управления нейтрализаций неисправных процессорных элементов K = $\{K_1, K_2, \ldots, K_6\}$ генерируются схемой формирования сигналов контроля (СФСК) таким образом, что Ki = 1, если i-й процессорный элемент исправен и Ki = 0 в противном случае. Сигнал Ki, поступающий на процессорный элемент Pi и соответствующий ему триггер T, управляет переключением их в третье состояние. Сигналы контроля процессорных элементов можно формировать различными способами. Один из возможных вариантов организации СФСК приведен на рис. 6.3. При этом предполагается, что средства отказоустойчивости могут быть ориентированы как на нейтрализацию производственных отказов с целью увеличения съема кристаллов с пластины, так и на нейтрализацию эксплуатационных

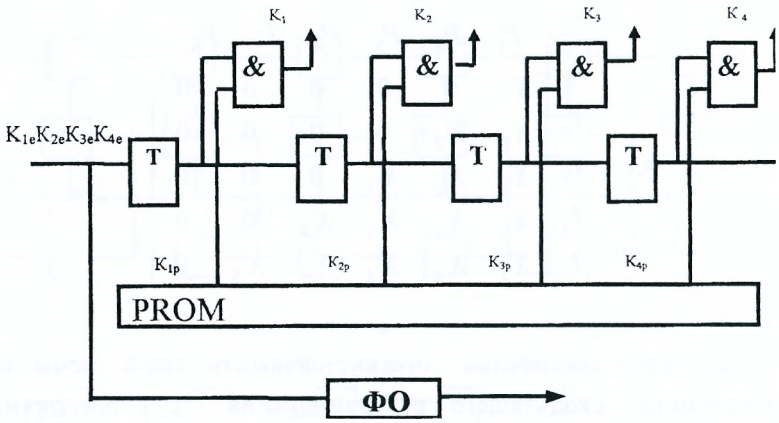


Рис. 6.3. Схема формирования сигналов контроля : ФО - схема определения фатального отказа.

отказов для улучшения показателей надежности схемы. Для нейтрализации производственных отказов в схему вводится ПЗУ дефектов (PROM) размерностью 1x(N + R), где N-количество основных, а R-количество резервных элементов схемы. При этом ПЗУ дефектов представляет собой набор плавких перемычек, путем пережигания которых на этапе производства в него заносится информация $K_p = (K_{1p}, K_{2p}, \ldots)$ о работоспособности соответствующих процессорных элементов. С целью нейтрализации эксплуатационных отказов для каждого процессорного элемента в СФСК вводится триггер работоспособности, в который заносится соответствующая информация о работоспособности процессорного элемента при тестировании схемы на этапе эксплуатации. Здесь предполагается, что проектируемая схема может находиться в следующих состояниях: нормальной работы, тестировании и реконфигурации. При этом имеется внешнее тестирующее устройство,

которое после тестирования схемы заносит соответствующую информацию о состоянии процессорных элементов $K_e = \{K_{1e}, K_{2e} \ldots\}$ в триггера работоспособности для осуществления реконфигурации схемы. Таким образом СФСК, на основе информации о тестировании схемы на этапе эксплуатации и производства, формирует сигналы контроля $K_1 = K_{1e}\ \&K_{1p}$, $K_2 = K_{2e}\ \&K_{2p}$ и т.д., которые управляют коммутационными элементами схемы, в результате чего реализуется заданный алгоритм реконфигурации.

При этом, в случае отказа процессорного элемента Pi, функции выполняемые процессорными элементами P_i, P_{i+1}, ..., P_N, передаются соответственно P_{i+1}, P_{i+2}, ..., P_{N+1}, т.е. осуществляется сдвиг вправо на один элемент. Приведенная на рис. 6.2б схема - тестопригодна, так как при помощи управления триггерами работоспособности СФСК можно легко проверить каждый из процессорных элементов в отдельности и транспортировать его внутреннее состояние на внешние выводы схемы, т.е. в предлагаемой схеме реализуется понятие управляемости и наблюдаемости [117]. Алгоритм тестирования при этом будет состоять из следующих шагов:

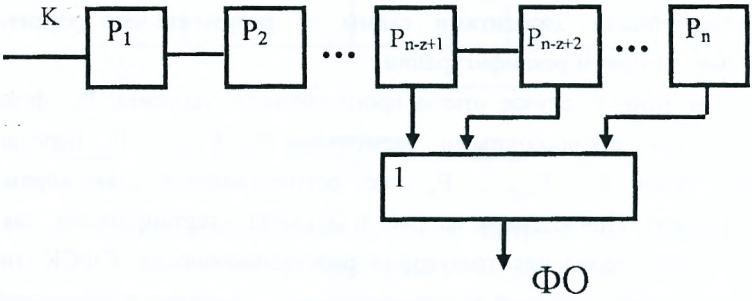
1. Все триггеры работоспособности процессорных элементов устанавливаются в нулевое состояние.

2. Путем задания соответствующей последовательности сигналов К устанавливаем триггер работоспособности процессорного элемента i, который мы хотим проверить в единичное состояние, а все остальные триггеры в нулевое состояние. При этом входные сигналы схемы поступают на i-й процессорный элемент, а выходные его сигналы являются выходными сигналами систолического процессора.

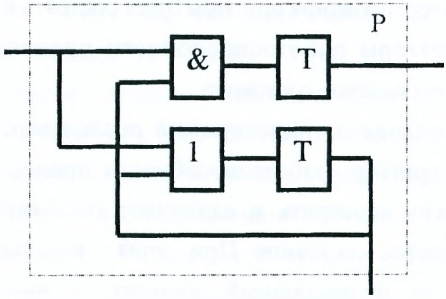
3. Производим тестирование i-го процессорного элемента и повторяем процесс начиная с пункта 2 для остальных процессорных элементов.

Если заранее не подготовлены тестовые последовательности, то

предлагаемая схема позволяет сама готовить для себя тестовые последовательности и эталонные реакции на них используя, например, мажоритарный принцип их формирования. В этом случае первый тестовый набор входных сигналов подается вначале, например, на первый процессорный элемент и фиксируется реакция на него, а затем на второй и на третий процессорные элементы. По мажоритарному принципу



а



б

Рис. 6.4. Схема формирования сигнала фатального отказа (а) и структура ее процессорного элемента (б) : ФО - сигнал фатального отказа.

(большинству сигналов на выходе) формируется эталонная реакция на первый тестовый набор. Затем подается второй тестовый набор и процесс повторяется.

В случае необходимости в отказоустойчивый систолический процессор может вводится также схема определения фатального отказа (**ФО**) как показано на рис.6.2б и рис. 6.3. Эта схема представляет собой одноразрядный систолический процессор сортировки с последовательным потоком входных данных, изображенный на рис. 6.4а. Схема процессорного элемента изображена на рис. 6.4б. Сигналы установки триггеров работоспособности синхронно подаются на схему определения фатального отказа и сортируются ей таким образом, что l сигналов неисправности схемы (последовательность $K_e = \{ K_{1e}, K_{2e} \dots \}$ имеет l нулей) появляются на l –крайних процессорных элементах схемы. Если Z – минимальное количество неисправных процессорных элементов схемы при котором идентифицируется состояние фатального отказа, то выходы Z – крайних правых процессорных элементов необходимо подключить к схеме “ИЛИ”, как показано на рис. 6.4а. Рассмотрим теперь учет аппаратных издержек на обеспечение приведенного метода обеспечения отказоустойчивости.

Общее количество ключей SW, добавляемых в схему, состоит из (N + R)b, где b-разрядность линии связи между процессорными элементами, одноразрядных ключей для обхода процессорных элементов по промежуточным данным, (N + R)b-одноразрядных ключей, добавляемых в процессорные элементы для управления переключением их в третье состояние, (N+R) – одноразрядных ключей для обхода триггеров управления и (N+R) – одноразрядных ключей для управления переключением соответствующих триггеров в третье состояние. Таким

образом, общее количество одноразрядных ключей SW, добавляемых в схему

$$N_{sw} = (N + R) 2 (b + 1). \qquad (6.19)$$

Аппаратные издержки на СФСК состоят из $(N + R)_e$ - плавких перемычек (ПЗУ-дефектов), $(N + R)_i$ - триггеров работоспособности и $(N + R)_u$ - двухвходовых элементов "И", т.е. общее количество добавляемых элементов в СФСК

$$N_k = (N + R)_e + (N + R)_i + (N + R)_u \qquad (6.20)$$

Затраты на схему формирования фатального отказа (ФО) состоят из $(N + R)_p$-одноразрядных процессорных элементов , изображенных на рис 6.4 и R_1-двухвходовоых схем " ИЛИ ". Отсюда общее количество элементов в схеме ФО

$$N_f = (N + R)_p + R_1, \qquad (6.21)$$

где $R_1 \equiv R$.

При этом дополнительные аппаратные издержки на один процессорный элемент без учета схемы определения фатального отказа состоят из 2(b + 1)-ключей SW, одной плавкой перемычки, одного триггера работоспособности и одного элемента "И", что намного меньше чем сложность процессорного элемента. Кроме этого в схему добавляется два внешних входа для управления триггерами работоспособности (информационный и синхронизации), а также один внешний вывод для сигнализации о фатальном отказе.

6.4.1 Уровень структурной перестройки схемы

Рассмотрим как влияет уровень резервирования на дополнительные аппаратные издержки для описанного выше метода обеспечения отказоустойчивости. Очевидно, что в качестве минимального уровня структурной перестройки схемы целесообразно рассматривать уровень простейших процессорных элементов, которые назовем атомарными. Пусть

а-уровень обеспечения отказоустойчивости в схеме, т.е. количество атомарных процессорных элементов в одном резервируемом элементе схемы. Предположим, что при а=1 схема состоит из N основных и R резервных элементов. Тогда в общем случае количество элементов на а-м уровне резервирования

$$N_R(a) = (N + R) / a = N(a) + R(a), \qquad (6.22)$$

где N(a), R(a) - соответственно количество основных и резервных элементов схемы на а-м уровне резервирования. Общее количество добавляемых в схему ключей SW:

$$N_{SW}(a) = (N + R)2(\ b + 1\ )/a = (N(a) + R(a))2(b + 1). \quad (6.23)$$

Аналогичным образом определяются затраты на СФСК и схему фатального отказа при изменении уровня резервирования. Как следует из приведенных выражений, с увеличением уровня резервирования схемы пропорционально уменьшаются аппаратные издержки на обеспечение отказоустойчивости и соответственно будет увеличиваться выход годных для области кристалла занятой этими схемами. Выход же годных для области кристалла, занятой резервируемыми элементами схемы уменьшается с увеличением уровня резервирования, так как уменьшается число разрешенных сочетаний [47] элементов схемы. От соотношения изменения выхода годных для различных областей кристалла и будет зависеть оптимальный уровень резервирования. Таким образом, задача определения оптимального уровня обеспечения отказоустойчивости состоит в разбиении схемы на такие подсхемы, при резервировании на уровне которых съем кристаллов с пластины будет максимальным. Определим общее количество возможных уровней обеспечения отказоустойчивости схемы. Исходя из принципа однородности схемы, каждый уровень резервирования должен состоять из идентичных

модулей схемы. Пусть N - целое число и

$$N = S_1^{a_1} S_2^{a_2} \ldots S_n^{a_m} \qquad (6.24)$$

- его разложение на простые множители. Тогда общее количество возможных уровней обеспечения отказоустойчивости V(N), будет равняться общему числу его положительных делителей и согласно [52] определяется следующим образом:

$$V(N) = (a_1 + 1)(a_2 + 1) \ldots (a_n + 1). \qquad (6.25)$$

6.4.2 Секционированное резервирование

Предположим, что схема разбита на секции и для каждой секции существуют свои резервные элементы, которые в случае необходимости могут заменить любой из элементов данной секции. Обозначим число секций через Nc, а число элементов в секции - N_0. Тогда N_0 характеризует степень распределяемости резервных элементов в схеме [47], которая определяет метод обеспечения отказоустойчивости. Так при $N_0 = 1$ для элементов схемы используется дублирование, при $N_0 = N$ - скользящее резервирование, а при $N_0 \neq 1$ и $N_0 \neq N$ согласно принятой терминологии - секционированное резервирование с различной степенью распределяемости резервных элементов схемы. Исходя из принципа обеспечения идентичности секций, общее количество возможных методов обеспечения отказоустойчивости $H_1(a)$ на фиксированном уровне резервирования с учетом трехкратного мажоритарного резервирования для линейных однородных схем

$$H_1(a) = V(N/a) + 1. \qquad (6.26)$$

Определим теперь общее количество возможных методов резервирования с учетом возможных уровней обеспечения

отказоустойчивости.

Теорема 6.2. Общее количество возможных методов обеспечения отказоустойчивости для линейных однородных схем определяется на основе следующего выражения:

$$H = V(N) + \sum_{a=1}^{V(N)} V(N/a), \qquad (6.27)$$

где a пробегает только те значения, которые соответствуют положительным делителям числа N.

Доказательство. Количество методов обеспечения отказоустойчивости на a-м уровне резервирования определяется согласно выражения (1). Так как общее число возможных уровней резервирования равно V(N), то общее количество методов обеспечения отказоустойчивости

$$H = \sum_{a=1}^{V(N)} V(N/a) + 1 = V(N) + \sum_{a=1}^{V(N)} V(N/a),$$

что и требовалось доказать.

Теорема 6.3. Если в каждой секции существует одинаковое количество резервных элементов, то аппаратные издержки на обеспечение отказоустойчивости при параллельном однородном потоке входных данных являются индифферентными к методу обеспечения отказоустойчивости и следовательно скользящее резервирование является оптимальным методом обеспечения отказоустойчивости.

Доказательство. Количество SW-ключей, необходимых для обхода неисправных элементов в одной секции,

$$N_{SC} = (No+Rc)2(b + 1).$$

Тогда общее количество ключей в схеме

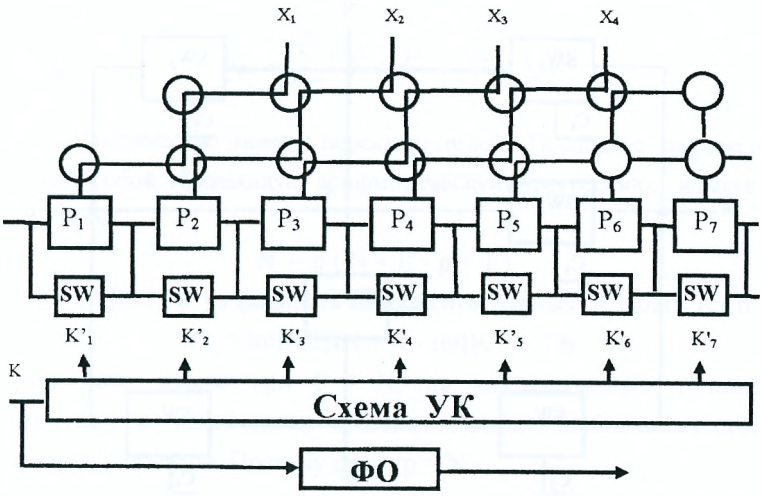
$$Ns = NscNc = (N + R)2(b + 1),$$

что совпадает с соответствующим выражением для скользящего резервирования. То же самое можно показать для СФСК и схемы фатального отказа. Таким образом, аппаратные издержки идентичны для различных методов резервирования. Так как число разрешенных сочетаний для скользящего резервирования при прочих равных условиях больше чем для других методов, то скользящее резервирование в данном случае является оптимальным методом обеспечения отказоустойчивости. Теорема доказана.

Схемы, построенные по предложенному выше методу, полностью самореконфигурируемы и обладают свойством самоорганизации. Это вытекает из того, что в таких схемах в случае необходимости возможен обход любого количества произвольных процессорных элементов, что позволяет создавать в рамках структуры схемы подструктуры произвольной размерности не превышающей размерности общей структуры схемы. При этом дополнительные аппаратные издержки незначительны. Рассмотренный выше подход может применяться также для адаптации нейронной сети к размерности решаемой задачи.

6.5. Параллельный неоднородный поток входных данных

Для обхода неисправных процессорных элементов при таком потоке входных данных применяются линейки переключателей, возможные состояния которых показаны на рис. 6.5б. Фрагмент отказоустойчивой схемы изображен на рис. 6.5а. Такая схема характеризуется тем, что поток данных X_i, позиционно расположен со сдвигом относительно того процессорного элемента, на который он поступает.



а

б

Рис. 6.5. Обеспечение отказоустойчивости при параллельном однородном потоке входных данных : **а** – схема обеспечения отказоустойчивости; **б** - возможные состояния переключателей

Схема управления коммутаторами (УК) на основе сигналов контроля поступающих от СФСК формирует сигналы управления переключателями таким образом, что они реализуют заданный алгоритм реконфигурации. Если Xi имеет один разряд, то структура переключателя состоит из шести одноразрядных ключей SW с тремя состояниями рис. 6.6.

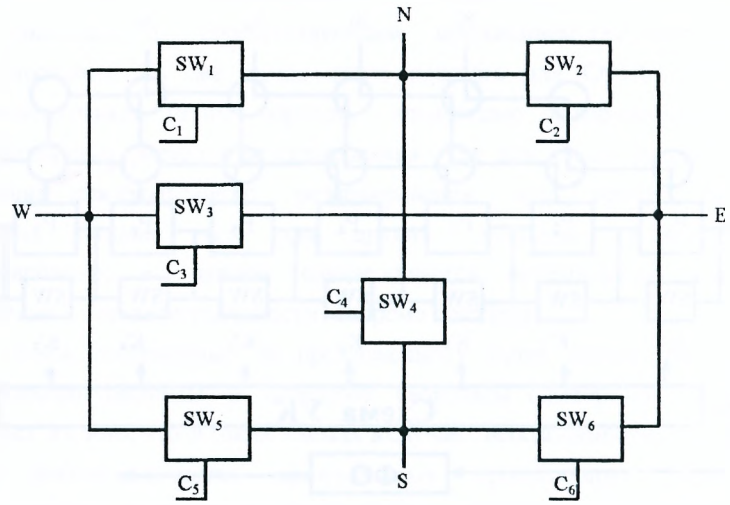


Рис. 6.6. Схема переключателя: C_1–C_6 – сигналы управления

Утверждение 6.1. Общее количество переключателей схемы (рис. 6.5), необходимых для нейтрализации любых R процессорных элементов определяется следующим образом:

$$N_p = int[R/2](N + R - int[R/2]+ 1)b,$$

если p<n и

$$N_p = N(R + 1)b, \text{ при } p = N,$$

где int - наибольшее ближайшее целое. При этом общее количество линеек переключателей

$$p = int[R/2] \quad \text{для} \quad R < 2N \quad \text{и} \quad p = N \quad \text{при} \quad R \ ?2N.$$

Доказательство. Как видно из рис. 6.5а каждая последующая линейка переключателей имеет на два переключателя меньше, чем предыдущая.

Исходя из этого

$$N_p = \sum_{i=1}^{p} (N + R - 2(i-1)),$$

где p - общее количество линеек переключателей. Последнее выражение представляет собой убывающую арифметическую прогрессию, исходя из свойств которой

$$N_p = p\,(N + R - p + 1).$$

Так как p = int[R/2,] а разрядность компоненты данных X_i равняется b, то

$$N_p = \text{int}[R/2](N + R - \text{int}[R/2] + 1)b,$$

что справедливо только при P < N, так как количество N-линеек переключателей в такой схеме позволяет обходить любое количество i-процессорных элементов. Поэтому при p = N

$$N_p = N\,(R + 1)\,b.$$

Рассмотрим некоторые вопросы построения таких схем. Введем координатную ось параллельную линейке процессорных элементов. За базовые координаты оси выберем номера процессорных элементов P_j, j = $\overline{1, N+R}$, расположенные в виде возрастающей последовательности. Данные X_i, i = $\overline{1, N}$, также упорядочим по номеру i в виде возрастающей последовательности с левого конца процессорной линейки. Координаты всех элементов схемы будем рассматривать относительно базовых координат P_j. Если X_i имеет координату P_j, то обозначим это как $X_i(j)$, а переключатель расположенный в m-м ряду и имеющий координату по оси абсцисс P_j- K(m, j). Тогда общий алгоритм построения отказоустойчивых схем в зависимости от объема резервных элементов будет состоять из следующих шагов:

1. В исходную схему добавляется P процессорных элементов и p = int[R/

2] линеек переключателей, причем количество переключателей в i-й линейке

$$N_{pi} = (N + P - 2(i - 1))b.$$

2. Входной поток данных X_p, $i = \overline{1, N}$, располагаем таким образом, что он имест координату X_i (p + i) и подключается непосредственно к переключателю с координатами K (p, p + i). Так для схемы, изображенной на рис. 6.4а с R = 4, X_i(2 + i) подключается к переключателю с координатами K (2, 2 +j).

3. Подключаем X_p, $i = \overline{1, N}$, к процессорному элементу P_p, путем соответствующей установки переключателей с координатами K(p, p + i), K(p,p +i - 1),K(p - 1,p+i - 1),K(p - 1,p + i - 2), ..., K(1, i), где общее количество таких переключателей для X_i равнятся 2p.

4. В исходную структуру схемы добавляется СУК и по аналогии с параллельным однородным потоком входных данных СФСК, схема определения фатального отказа и 2(N+ R)b одноразрядных ключей SW для обхода процессорных элементов по промежуточным данным.

Таким же способом, но не наоборот, можно обеспечить обход неисправных процессорных элементов при параллельном однородном потоке входных данных. Однако с учетом того, что каждый переключатель содержит шесть простых ключей SW, легко показать, что с точки зрения аппаратных издержек на обеспечение отказоустойчивости это является нерациональным.

6.5.1 Секционированное резервирование

Пример обеспечения данного метода отказоустойчивости изображен на рис. 6.7, где R_{11}, R_{12} - резервные элементы первой секции, а R_{21}, R_{22} - резервные элементы второй секции. В общем случае число линеек

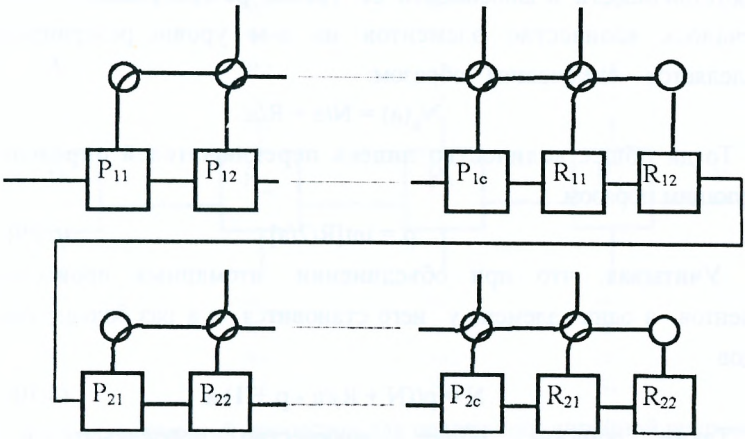


Рис. 6.7. Секционированный метод обеспечения отказоустойчивости

переключателей для каждой секции определяется следующим образом:

$$p = int[Rc/2] = int\ [R / Nc / 2],$$

что в общем случае в Nc раз меньше чем для скользящего резервирования.

Общее количество переключателей при этом в предположении, что в каждой секции существует одинаковое количество резервных элементов:

$$N_p = N_c N_{pc} = N_c p(N_0 + R_c - p + 1) = p\,(N + R - N_c p + N_c), \quad (6.28)$$

где N_{pc} – число переключателей в секции. Это меньше чем для скользящего резервирования. Отсюда следует, что чем больше количество секций, тем меньше аппаратных издержек на обеспечение отказоустойчивости при одном и том же объеме резервных элементов. Поэтому для данного метода обеспечения отказоустойчивости необходимо определять оптимальную степень распределяемости резервных элементов.

6.5.2 Уровень обеспечения отказоустойчивости

Рассмотрим как меняются аппаратные издержки на обеспечение

отказоустойчивости в зависимости от уровня резервирования. Как уже отмечалось количество элементов на a-м уровне резервирования определяется следующим образом:

$$N_R(a) = N/a + R/a.$$

Тогда общее количество линеек переключателей определяется следующим образом:

$$p = int[R/2/a]. \qquad (6.29)$$

Учитывая, что при объединении атомарных процессорных элементов в один элемент у него становится в a раз больше верхних входов

$$N = p((N + R)/a - p + 1)\, a. \qquad (6.30)$$

Таким образом, общее количество переключателей при увеличении уровня резервирования a уменьшается, поэтому здесь как и для параллельного однородного потока входных данных необходим поиск оптимального уровня обеспечения отказоустойчивости.

В заключение отметим, что обеспечение отказоустойчивости структур с последовательным потоком входных данных производится также как обход неисправных процессорных элементов по промежуточным данным в рассмотренных структурах.

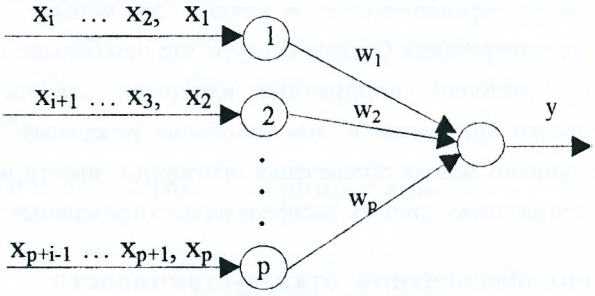


Рис. 6.8. Линейная нейронная сеть

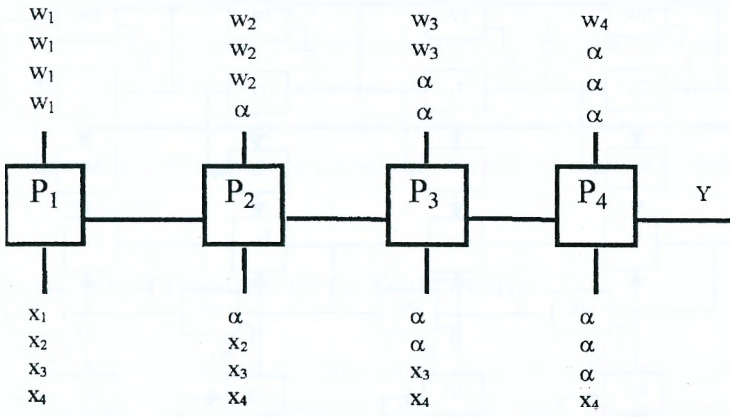


Рис. 6.9. Систолический процессор для реализации линейной нейронной сети

6.6. Самоорганизующаяся линейная нейронная сеть

Рассмотрим обеспечение структурной самоорганизации нейронной сети для решения задач прогнозирования. Такая сеть представлена на рис. 6.8. Она состоит из выходного нейронного элемента с линейной функцией активации и p распределительных нейронных элементов, где p – размерность окна (глава 2 тома 1). Схема систолического процессора и организация потоков входных данных для линейной сети представлены на рис. 6.9.

Как следует из рисунка, потоки входных данных X и весовых коэффициентов W являются параллельными однородными. Тогда обеспечение структурной самоорганизации систолического процессора буде происходить в соответствии с разделом 6.5. Схема самоорганизующегося процессора представлена на рис. 6.10. Для управления занесением входных данных X в регистры RG_2 предназначена цепочка триггеров T_2. Если поток входных $X = (x_1,x_2,\ldots,x_4)$, а сигналы управления $U_2 = (1,1,1,1)$, то в первом

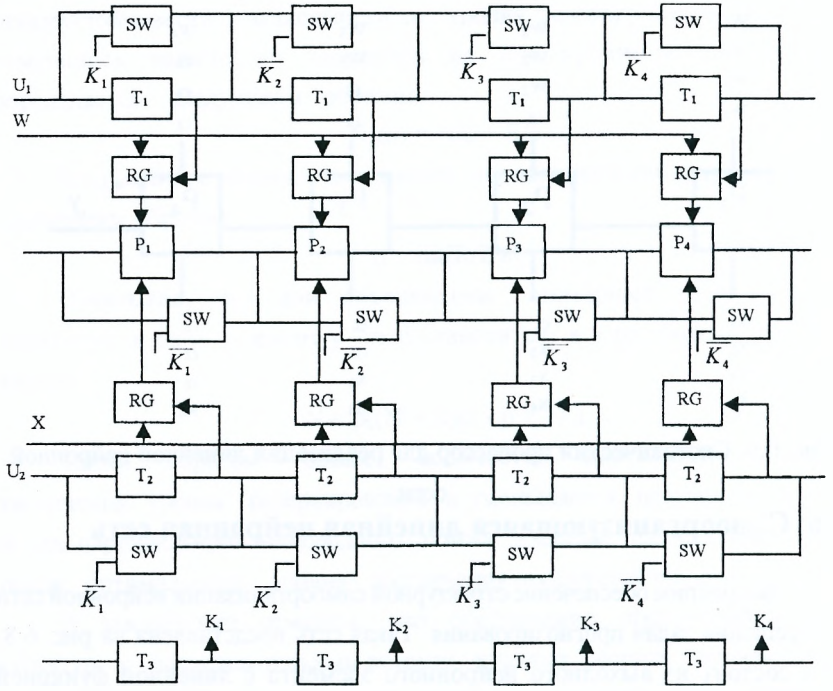


Рис. 6.10. Самоорганизующийся систолический процессор

такте в регистр RG_2 процессорного элемента P_1 заносится компонента x_1, во втором такте в регистры RG_2 процессорных элементов P_1 и P_2 запишется компонента x_2 и т. д. Цепочка триггеров T_1 предназначена для управления записью весовых коэффициентов W в регистры RG_1. Если поток весовых коэффициентов $W = (w_1, w_2, \ldots, w_4)$, то сигналы управления U_1 формируются следующим образом:

$$U_1 = (1,0,\ldots,0)$$

Ключи SW предназначены для обхода соответствующих элементов схемы. Сигналы управления ключами K формируются триггерами T_3 (рис.

6.10). Путем записи в триггеры T_3 соответствующей последовательности сигналов $K = (K_1, K_2, \ldots)$ можно изменять размерность систолического массива и обеспечить необходимую конфигурацию схемы. В результате этого реализуется структурная самоорганизация схемы, которая может применяться как для адаптации к размерности решаемой задачи, так и для обеспечения отказоустойчивости.

6.7. Отказоустойчивая сеть Хопфилда

Рассмотрим применение описанных выше методов на примере проектирования отказоустойчивого систолического процессора для реализации нейронной сети Хопфилда на кристалле. В матричной форме модель сети Хопфилда можно представить следующим образом:

$$y(t+1) = F(S(t)),$$

$$S(t) = W^T Y(t),$$

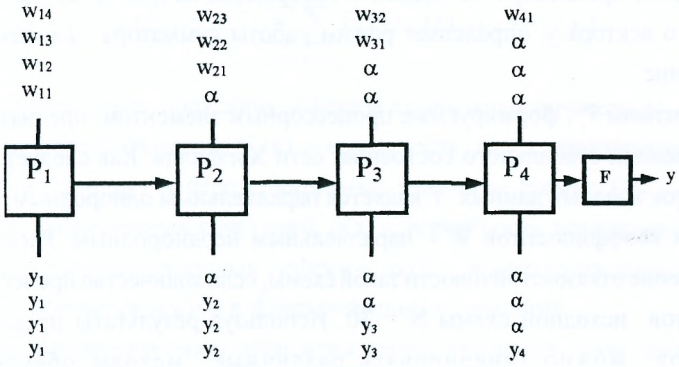


Рис. 6.11. Реализация сети Хопфилда на систолическом процессоре

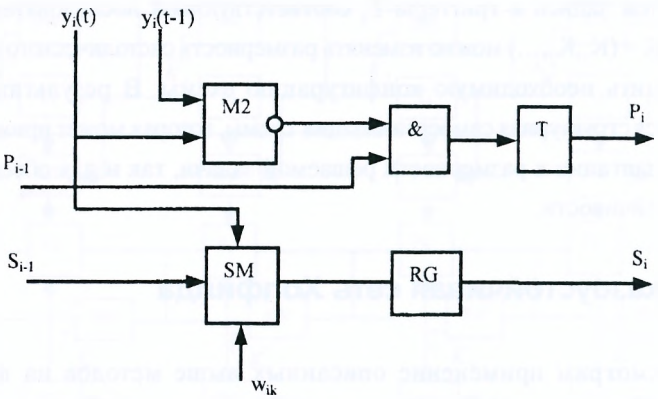


Рис. 6.12. Схема процессорного элемента

где S(t) – взвешенная сумма.

Пусть элементы вектора Y являются биполярными, т.е. $y_i \in [1,-1]$. Систолический процессор для реализации сети Хопфилда и организация потоков входных данных представлеы на рис. 6.11.

Схема процессорного элемента изображена на рис. 6.12, где элемент входного вектора y_i определяет режим работы сумматора : сложение или вычитание.

Сигналы P_i, формируемые процессорным элементом предназначены для фиксации стабильного состояния сети Хопфилда. Как следует из рис. 6.11 поток входных данных Y является параллельным однородным, а поток весовых коэффициентов W – параллельным неоднородным. Рассмотрим обеспечение отказоустойчивости такой схемы, если количество процессорных элементов исходной схемы N = 30. Используя результаты предыдущих разделов можно генерировать различные методы обеспечения отказоустойчивости для данной схемы. Тогда общее количество методов

обеспечения отказоустойчивости такой схемы согласно теореме 6.2:

H = V(30)+V(30)+V(15)+V(10)+V(6)+V(5)+V(3)+V(2)+V(1) = 35.

Рассмотрим обеспечение отказоустойчивости процессора Хопфилда на уровне атомарных процессорных элементов, т.е. a = 1. В этом случае количество возможных методов отказоустойчивости без учета мажоритарного резервирования:

$$H_1(1) = V(30) = 8.$$

При проектировании на уровне кристалла задача синтеза отказоустойчивой схемы состоит в выборе такого метода и объема резервных элементов, чтобы обеспечить максимальное значение показателя качества резервирования [43]

$$\max\{F(R)\}$$

при ограничении, например, по площади кристалла.

Для расчета выхода годных процессора Хопфилда будем использовать следующую модель:

$$Y(R) = p_1^{N_P} \sum_{x=0} \sum_{i=0}^{R} k(i) p_x(i) p(x) p_m(x),$$

где p_1 – вероятность не попадания дефектов в один переключатель; N_p – общее количество переключателей; $p_m(x)$ – вероятность, что x производственных дефектов не попадут в нерезервируемые элементы схемы; k(i) – общее количество разрешенных сочетаний; p(x) – вероятность возникновения на кристалле x производственных дефектов; $p_x(i)$ – вероятность попадания x производственных дефектов в фиксированные i элементов.

К нерезервируемым элементам схемы относятся такие элементы, попадание дефектов в которые приводит к катастрофическому отказу. Для процессора Хопфилда – это ключи SW, переключатели, блок нелинейного

преобразования F и соединения между элементами. Определим вероятность $P_m(x)$. Число распределений дефектов, которые оставляют пустыми не резервируемые элементы схемы, равняется

$$G_1 = (A_1(N+R))^x,$$

где A_1 – площадь одного резервируемого элемента схемы.

Общее количество распределений дефектов по элементам схемы можно определить следующим образом:

$$G_2 = ((N+R)(A_1 + A_n))^x,$$

где A_n – площадь не резервируемого элемента схемы, приходящаяся на один процессорный элемент.

Тогда

$$p_m(x) = G_1 / G_2 = (A_1 /(A_1 + A_n))^x$$

Не претендуя на точность с целью моделирования выхода годных схемы предположим, что площадь одного резервируемого элемента на кристалле составляет $A_1 = 0.015 см^2$, а площадь одного нерезервируемого элемента – $A_n = 0.0001 см^2$. Пусть плотность дефектов $D = 8\ дф/см^2$, а параметр группирования дефектов на кристалле α=2. Предположим, что имеет место кластеризация дефектов на большой площади. Тогда в соответствии с разделом 6.1 для расчета выхода годных схемы будем использовать в качестве p(x) обобщенное отрицательное биноминальное распределение, а в качестве $p_x(i)$ – статистику Максвелла–Больцмана.

Выход годных исходной схемы можно приближенно определить

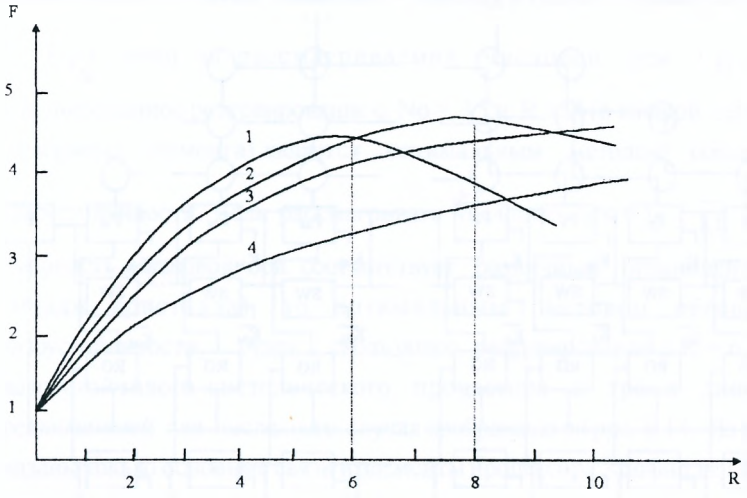


Рис. 6.13. Графики зависимостей показателя качества резервирования от объема резервных элементов для разных методов отказоустойчивости: 1 – скользящее резервирование, 2, 3, 4 - секционированное резервирование соответственно со степенью распределяемости резервных элементов 15, 10 и 6.

следующим образом

$$Y(0) = 1 / (1 + A_1 * 30\ D / \alpha)^{\alpha} = 0.1275.$$

На рис. 6.13 представлены графики зависимостей показателя качества резервирования от объема резервных элементов при различной степени распределяемости резервных элементов (No = 30, No = 15, No = 10 и No = 6). При этом предполагалось, что вероятность не попадания дефектов в один переключатель равняется 0.998. Из рисунка видно, что максимальное значение показателя качества обеспечения отказоустойчивости для скользящего резервирования достигается при R = 6 (выход годных при этом равняется 0.65), а для секционированного

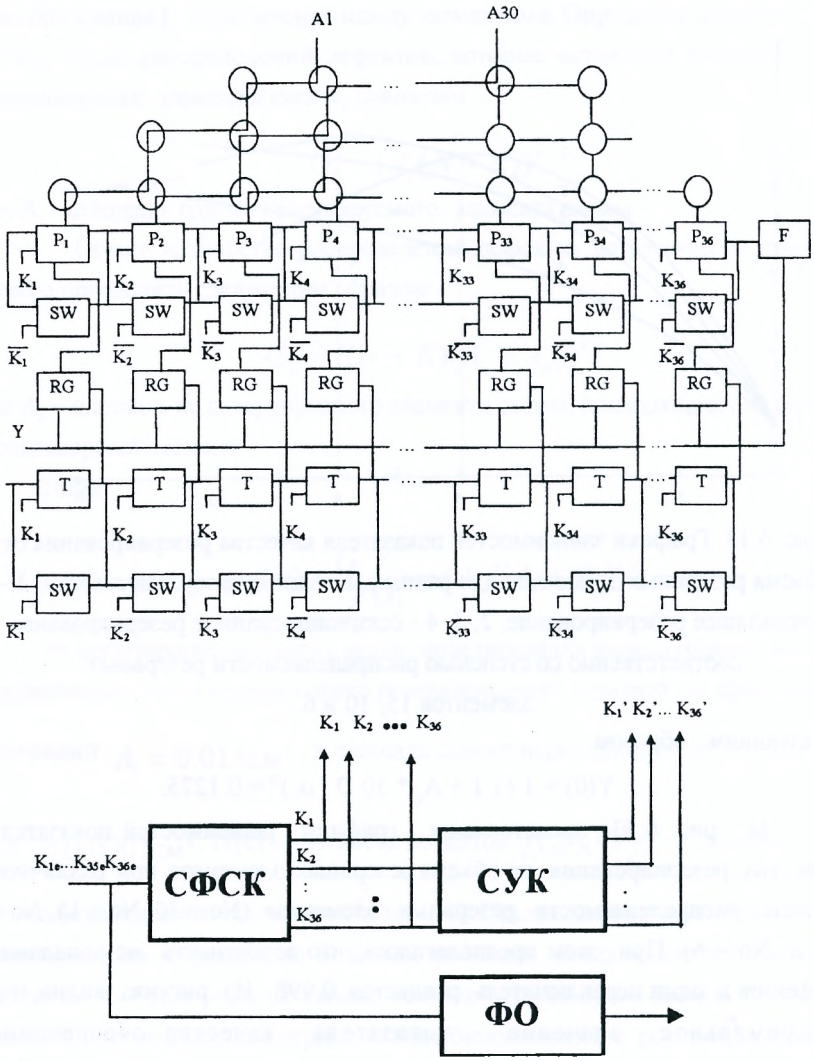


Рис. 6.14. Отказоустойчивый процессор Хопфилда: K_1',K_2',...,K_{36}' - сигналы управления переключателями

резервирования с двумя секциями - при R = 8 (выход годных составляет 0.7). При этом на рассматриваемой числовой оси ($R = \overline{1,10}$) секционированное резервирование с No = 15 и R = 8 (в каждой секции по 4 резервных элемента) является оптимальным методом обеспечения отказоустойчивости. Если рассматривать числовую ось с $R = \overline{1,6}$ (разная размерность числовой оси соответствует различным ограничениям по площади кристалла), то оптимальным методом обеспечения отказоустойчивости будет скользящее резервирование с R = 6. Схема отказоустойчивого систолического процессора с тремя линейками переключателей для последнего случая изображена на рис. 6.14. На рисунке показаны только основные связи и элементы процессора Хопфилда. При этом резервные элементы могут использоваться как для съема кристаллов с пластины, так и для улучшения показателей надежности схемы. Таким образом в данной главе рассмотрены методы структурной самоорганизации нейронных сетей, которые реализуются на систолических процессорах. Такие методы могут применяться как для обеспечения отказоустойчивости, так и для адаптации схемы к размерности решаемой задачи.

ЛИТЕРАТУРА

1. Kohonen T. Sell–organised formation of topologically correct feature maps// Biological Cybernetics. – 1982. – N43.–P.59–69.

2. Kohonen T. Self–organizing and Associative Memory. –Berlin: Springer Verlag.–1984.

3. Kohonen T. The self–organizing map // Proceedings of the IEEE.– 1990.– N.78– P.1464–1480.

4. Kohonen T. Self–organizing Maps.– Heidelberg: Springer Verlag.– 1995.

5. Ruediger Brause. Neuronale Netze eine Einfuehrung in die Neuroinformatik.–Stuttgart: Teubner.–1995–462p.

6. Grossberg S. Adaptive pattern classification and universal recoding // Biological Cybernetics – 1976. – N. 23.– P.121–134.

7. Kroese B. An introduction to Neural Networks.– Amsterdam: Univercity of Amsterdam.– 1996.–120p.

8. Grossberg S. Competetive learning: From interactive activation to adaptive resonance // Cognitive Science.– 1987.–N.11.– P.23–63.

9. Kohonen T., Hynninen J., Kandas J., Laaksonen J., Torkkola K. LVQPAK: The Learning Vector Quantization. – Helsinki: Universitety of Technology.– 1995.– 300p.

10. Grossberg S. Adaptive pattern classification and universal recoding // ing // Biological Cybernetics – 1976. – N. 23.– P.121–134.

11. Lipmann R. An introduction to computing with neural nets // IEEE Acoustic, Speech and Signal Processing Magazine.–1987.–N2– P.4–22.

12. Grossberg S. Nonlinear neural networks: principles, mechanisms and ; and architectures // Neural Networks.– 1988.–N.1.– P.17–61.

13. Grossberg S. Carpenter G. A Massively Parallel Arhitecture for a Self–

Organizing Neural Pattern Recognition Machine // Neural Networks and Natural Intelligence.– MIT Press.– 1988.–V.37.– P.54–115.

14.Rojas Raul. Theorie der neuronalen Netze. Eine systematische che Einfuehrung. – Berlin: Springer-Verlag. – 1993. – 446p.

15.Hecht–Nielsen R. Counterpropagation Networks // Proceedings of the the IEEE First International Conference on Neural Networks.– IEEE Press.–1987.– P.19 – 32.

16. Hecht–Nielson R. Applications of Counterpropagation Networks // Neural Networks.–1988.– V.1.–N2.– P.131–139.

17.Moody J., Darken C. Fast Learning in networks locally–tuned processing units // Neural Computation.– 1989.–N1.–P.281–294.

18.Bishop C. Neural Networks for Pattern Recognition.– Oxford University Press.–1995.

19.Hartman E., Keeler D., Kawalski J. Layered neural networks with gaussian hidden units as universal approximator // Neural Networks. – 1990.– V.35.– N2.– P.210–215.

20. Golovko V., Suhodolsky H.,Savitsky J., Gladischuk V. and Dimakov V. Neural Networks for optimization problems // Proceedings of Int. Conf. on design methodologies for signal processing. –Szczecin: Silesian Technical University.– 1996.–P.38–41.

21. Golovko V., Suhodolsky H., Dimakov V. Neural Net for combinatorial optimization // Proceedings of the High Performance Computing Simposium.– San Diego: The Society for Computer Simulation International.– 1998.

22.Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые ые задачи: Пер. с англ. – М.: Мир.– 1982.–416с.

23.Мину М. Математическое программирование. Теория и алгоритмы: ы: Пер. с фр. – М.: Наука. –1990.– 488с.

24.Millan J. Reinforcement learning of goaldirected obstacle–avoiding

reaction strategies in an autonomous mobile robot // Robotics and Autonomous Systems.– 1995. – V.15.–N.4.–P.273–297.

25. Thrun S. An approach to learning mobile robot navigation // Robotics otics and Autonomous Systems.– 1995.–V.15.–N4–P.301–319.

26. Jlg W., Berns R. A learning arhitecture based on reinforcement learning for adaptive control of the Walking machine LAURON // Robotics and Autonomous Systems.–1995.–V15.–N4–P.321–334.

27. Schilling K., Roth H., Sensordatefusion mit Fuzzy Logic zur Steuerung mobiler Roboter // Neuro–fuzzy symposium. – Friedrichshafen: Graf–Zeppelin–Haus.– 1995.

28. Schilling K., Garbajosa J.,Mellado M., Mayerhafer R. Design of Flexible Autonomes Transport Robots for Industrial Production // Proceedings IEEE International Simposium on Industrial Electronics.–Guimares. – IEEE Press.– 1997.–V.3.– P.791–796.

29. Roth H., Shilling K. Rotating Ultradonic Sensors for Obstacle tacle Classification Applied to Mobile Robots // Proceedings 4th International Conference on Mechatronics and Machine Vision in Practice.–1997.

30. Schilling K., Golovko V., Dimakov V. Neural system for mobile robot autonomous navigation // Proccedings of International Workshop on disign methodologies for signal processing.–Szczecin: Silesian Technical University.– 1996.– P.124–130.

31. Golovko V., Dimakov V., Savitsky Ju., Gladischuk V. Neural system for intelligent robot navigation // Proceedings of International Conference on Technical Informatics.– Timisoara: University of Timisoara.– 1996.–P.63–70.

32. Golovko V., Dimakov V. Neural Networks for autonomous mobile robot robot control// Proceedings of international Conference NITE'96.–Szczecin: Technical University. –1996.

33. Golovko V., Dimakov V. Schilling K. Intelectual system for control of

mobile robot // Proceedings of International Conference "New trends in Artifical Intelligence and Neural Networks".–Ankara: EMO Scientific Books.– 1997.

34. Golovko V., Dimakov V. Neural Control System for Mobile Robot // Preprints of International Workshop on Intellegent Control INCON'97: – Sofia: Techical University.–1997.

35. Golovko V., Dimakov V. Intellectual Simulation of Mobile Robot Control System // Proccedings of the High Perfomance Computing Symposium.–San Diego: The Society for Computer Simulation International.–1998.–P.440–445.

36. Golovko V., Dimakov V. Intelligent Neural System for Vehicle Control // Proceeding of the High Perfomance Computing Symposium.–San Diego: The Society for Computer Simulation International.–1998.–P.110–115.

37. Golovko V., Dimakov V. Arhitecture of Neural System for Control of Autonomous Vehicles // Preprints of the 3rd IFAC Symposium on Intelligent Autonomous Vehicles.–Oxford UK: Elsevier Science Ltd.– 1998.–V.1– P.287–297.

38. Головко В. Интелектуальная нейронная система для автономного управления мобильным роботом // Труды X научно–технической конференции .– Брест: БПИ.–1988.–С.15–25.

39. Golovko V., Dimakov V. Neural Systems for a Route Planning for the Mobile Robot // Proceedings of the Seventh Turkish Symposium on Artifical Intelligence and Neural Networks.– Ankara: Bilkent University.–1988.–P.267–268.

40. Jochem T., Pomerlau D., Thorpe C. MANIAC: A Next Generation Neurally Based Autonomous Road Follower // Proceedings of the International Conference on Intelligent Autonomous Systems.–Pittsburgh. 1993.

41. Pomerlau D., Gowdy J., Thorpe C. Combining Artifical Neural Networks and Symbolic Processing for Autonomous Robot Guidance // Proceedings of IEEE Conference on Intelligent Robots and Systems. Pittsburg.– 1995.–P.961–967.

42. Кун С. Матричные процессоры на СБИС: Пер. с англ.–М.: Мир.–1991.–672с.

43. Головко В. Методы обеспечения отказоустойчивости линейных систолических процессоров // Микроэлектроника.–1995.–Т.24.–N3.–С.229–240.

44. Стэппер Ч., Армстронг Ф., Саджи К. Статистические модели выхода годных интегральных микросхем// ТИИЭР.–1983.–Т.74.–N4.–С.6–26.

45. Stapper C. Large–area fault clusters and fault talerance in VLSI circuits: a review // IBM J. Res. Develop.–1989.–V.33.–N2.–P.162–173.

46. Головко В. Показатели эффективности отказоустойчивых структур СБИС.–Минск: ИТК АН РБ. – 1991.–N15.–20с.

47. Головко В. Статистические модели выхода годных для отказоустойчивых схем на кристалле // Микроэлектроника. – 1992.–Т.21.–N1.–С.20–26.

48. Головко В. Некоторые аспекты определения выхода годных для отказоустойчивых схем на кристалле // Микроэлектроника. – 1992.–Т.21.–N5.–С.37–44.

49. Golovko V. Self–organization of systolic processors // Proceedings of the International Conference CMNDT–95.–Berlin: DGZfP.–1995.

50. Головко В. Самоорганизующиеся линейные процессоры // Сборник трудов конференции "Распознавание образов и обработка информации".–Минск: ИТК АН РБ.– 1995.–С.82–87.

51. Ярмолик В. Контроль и диагностика цифровых узлов ЭВМ. – Минск: Наука и техника. – 1988.–240с.

52. Айерлэнд К., Роузен М. Классическое введение в современную теорию чисел. –М.: Мир. – 1987.–416с.

СОДЕРЖАНИЕ

Предисловие 3

Глава 1. Саморганизующиеся нейронные сети Кохонена 6

1.1. Общая характеристика сетей Кохонена 6
1.2. Конкурентное обучение 8
1.2. Векторный квантователь 17
1.3.1. Конкурентное обучение с одним победителем 18
1.3.2 Конкурентное обучение со многими победителями 20
1.3.3. Контролируемое конкурентное обучение 20
1.4. Самоорганизующиеся карты Кохонена 21
1.5. Решение задачи коммивояжера 29

Глава 2. Нейронные сети адаптивного резонанса 34

2.1. Основы адаптивного резонанса 34
2.2. Архитектура нейронной сети адаптивного резонанса 38
2.3. Функционирование ART1 сети 41
2.4. Алгоритм обучения и функционирования ART1 сети 44

Глава 3. Гибридные нейронные сети 48

3.1. Нейронные сети встречного распространения 48
3.1.1. Линейная аппроксимация функций поверхностями постоянного уровня 49
3.1.2. Линейная аппроксимация функций поверхностями переменного уровня 54
3.1.3. Нелинейная аппроксимация функций 57
3.2. Нейронные сети с радиально-базисной функцией 59
3.3. Иерархический классификатор 64
3.3.1. Архитектура нейронной сети 65
3.3.2. Обучение и функционирование 66
3.4. Решение задач оптимизации 68

3.4.1. Задача о кратчайшем пути 69
3.4.2. Задача о рюкзаке 75

Глава 4. Применение нейронных сетей для управления 82

4.1. Управление движением робота по заданной траектории 85
4.1.1. Общая структура системы 85
4.1.2. Архитектура нейронной сети 86
4.1.3. Обучение и формирование обучающей выборки 87
4.1.4. Тестирование 91
4.2. Автономное управление мобильным роботом 92
4.2.1. Входная информация и задачи нейронной системы 94
4.2.2. Архитектура нейронной системы 96
4.2.3. Формирование карты местности 99
4.2.4 Блок определения оптимального интервала движения 109
4.2.5. Аналитический блок 118
4.2.6. Многослойный персептрон 128
4.2.7. Бинарный блок 135
4.2.8. Самообучение и самоорганизация 143
4.2.9. Тестирование 148
4.3. Автономное управление автомобилем 151

Глава 5. Отображения нейронных сетей на систолические процессоры 155

5.1. Однослойные нейронные сети 156
5.1.1. Линейные систолические массивы 156
5.1.2. Матричные систолические процессоры 164
5.2. Многослойные нейронные сети 166
5.2.1. Линейное объединение 167
5.2.2. Матричное объединение 171
5.3. Релаксационные нейронные сети 174

Глава 6. Самоорганизация и отказоустойчивость 178

6.1 Статистические модели выхода годных 179
6.2. Показатели надежности 188
6.3. Факторы, влияющие на структурную реконфигурацию линейных

систолических процессоров 190
6.4. Параллельный однородный поток входных данных 193
6.4.1 Уровень структурной перестройки схемы 200
6.4.2 Секционированное резервирование 202
6.5. Параллельный неоднородный поток входных данных 204
6.5.1 Секционированное резервирование 208
6.5.2 Уровень обеспечения отказоустойчивости 209
6.6. Самоорганизующаяся линейная нейронная сеть 211
6.7. Отказоустойчивая сеть Хопфилда 213

Литература 220

Научное издание

Владимир Адамович Головко

НЕЙРОИНТЕЛЛЕКТ: ТЕОРИЯ И ПРИМЕНЕНИЕ
КНИГА 2
Самоорганизация, отказоустойчивость и применение нейронных сетей

Редактор	Т.В. Строкач
Художественный редактор	А.П. Дунец
Компьютерный дизайн и верстка	А.П. Дунец

Лицензия № 382 от 30.04.1999г. Подписано в печать 9.08.99. Формат 60x84/16. Бумага CopyRex. Гарнитура Times New Roman. Усл. печ. л. 13,2 Уч.-изд. л.14,3 Заказ № 495 Тираж 500 экз. Отпечатано на ризографе Брестского политехнического института. 224017, г. Брест, Московская, 267.