# Вычислительные машины, комплексы, системы и сети

## ч. 2

# Основные понятия

**Система обработки данных (информационная система)** - совокупность технических и программных средств, предназначенных для информационного обслуживания людей и технических объектов.

Классы информационных систем:

*вычислительные машины* (**ВМ**)

*вычислительные системы* (**ВС**)

*вычислительные комплексы* (**ВК**)

*сети*

**ВМ** предназначены для решения широкого круга задач пользователями, работающими в различных предметных областях. Основной блок ВМ - процессор. Процессор инициализирует процесс исполнения программы и управляет им.

**ВК** - это несколько **ВМ**, информационно связанных между собой. При этом каждая ВМ самостоятельно управляет своими вычислительными процессами. Информационный обмен между ВМ комплекса менее интенсивен (в сравнении с информационным взаимодействием процессоров в мультипроцессорных системах). Широкое применение ВК получили в информационно-управляющих системах.

# Основные понятия

**ВС** - это информационная система, настроенная на решение задач конкретной области применения, т.е. в ней имеется аппаратная и программная специализация. Часто ВС содержит несколько процессоров, между которыми в процессе работы происходит интенсивный обмен информацией, и которые имеют единое управление вычислительными процессами. Такие системы называются *мультипроцессорными.* Другим распространенным типом ВС являются *микропроцессорные системы.* Они строятся использованием либо микропроцессора (МП), либо микроконтроллера, либо специализированного процессора цифровой обработки сигналов.

Три метода, обеспечивающих увеличение производительности систем:

- совершенствование элементной базы,
- структурные методы
- математические методы.

**Параллельные вычислительные системы** - это физические компьютерные, а также программные системы, реализующие тем или иным способом параллельную обработку данных на многих вычислительных узлах.

# Классификация Флинна

Классификация базируется на понятии **потока, под которым понимается последовательность элементов, команд или данных, обрабатываемая процессором.**

Четыре класса архитектур: **SISD, MISD, SIMD, MIMD**.

***SISD*** (single instruction stream / single data stream) - одиночный поток команд и одиночный поток данных. К этому классу относятся классические последовательные машины (машины фон-неймановского типа). В таких машинах есть только один поток команд, все команды обрабатываются последовательно друг за другом, и каждая команда инициирует одну операцию с одним потоком данных.

***SIMD*** (single instruction stream / multiple data stream) - одиночный поток команд и множественный поток данных. Поток команд, включает, в отличие от **SISD**, векторные команды. Это позволяет выполнять одну арифметическую операцию сразу над многими данными - элементами вектора.

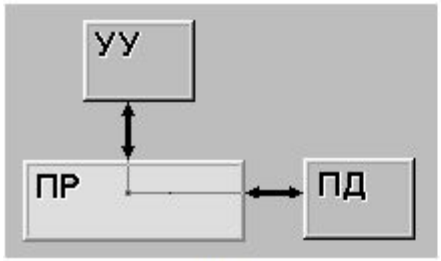
***MISD*** (multiple instruction stream / single data stream) - множественный поток команд и одиночный поток данных. Определение подразумевает наличие в архитектуре многих процессоров, обрабатывающих один и тот же поток данных. Реально существующей вычислительной системы, построенной на данном принципе, пока нет.

***MIMD*** (multiple instruction stream / multiple data stream) - множественный поток команд и множественный поток данных. В вычислительной системе есть несколько устройств обработки команд, объединенных в единый комплекс и работающих каждое со своим потоком команд и данных (мультипроцессорные системы).

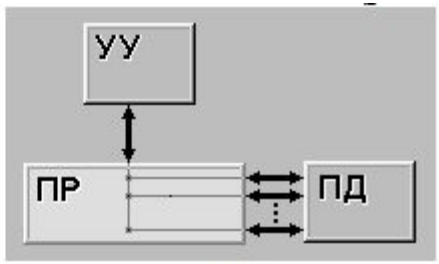
# Классификация Флинна

Четыре класса архитектур:

**SISD**



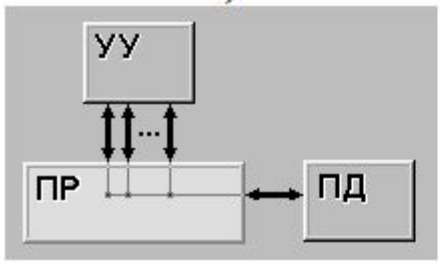

а)



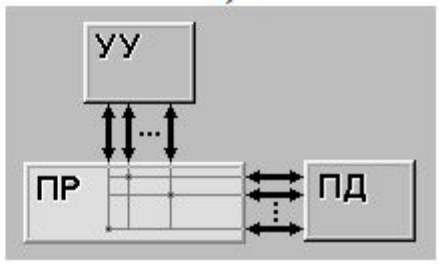

б)

**SIMD**

**MISD**




в)




г)

**MIMD**

# Вычислительные системы класса SIMD

Модель вычислений: **одиночная операция выполняется над большим блоком данных.**

Два типа: векторно-конвейерные и матричные системы.

**Векторно-конвейерные системы (PVP компьютеры)**

Два принципа в архитектуре процессоров:

конвейерная организация обработки потока команд

введение в систему команд набора векторных операций, которые позволяют оперировать с целыми массивами данных

**Векторные команды оперируют целыми массивами независимых данных, т. е. команда вида A=B+C означает сложение двух массивов, а не двух чисел.**

Конвейеризация эффективна только тогда, когда загрузка конвейера близка к полной, а скорость подачи новых операндов соответствует максимальной производительности конвейера. Векторные операции обеспечивают идеальную возможность полной загрузки вычислительного конвейера.

При выполнении **векторной команды** одна и та же операция применяется ко всем элементам вектора

# Вычислительные системы класса SIMD

В векторно-конвейерной системе имеется несколько один или несколько конвейерных процессоров, выполняющих векторные команды путем засылки элементов векторов в конвейер с интервалом, равным длительности прохождения одной, стадии обработки.

Векторная обработка увеличивает скорость и эффективность обработки за счет того, что обработка целого набора (вектора) данных выполняется одной командой.

Векторные процессоры должны иметь гораздо более сложную структуру и, по сути дела, **содержать множество арифметических устройств**

Типовая организации векторной вычислительной системы включает в себя:

блок обработки команд, осуществляющий вызов и декодирование команд,

векторный процессор, исполняющий векторные команды,

скалярный процессор, исполняющий скалярные команды,

память для хранения программ и данных.

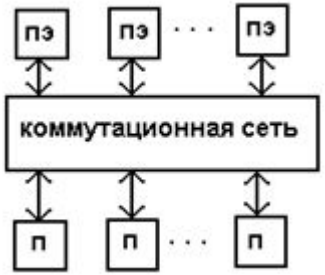
Длина одновременно обрабатываемых векторов в современных векторных компьютерах составляет, как правило, 128 или 256 элементов.

# Матричные системы

Матричные системы лучше всего приспособлены для решения задач, характеризующихся ***параллелизмом*** независимых объектов или *данных*.

Матричная система состоит из множества **процессорных элементов** (ПЭ), организованных таким образом, что они исполняют векторные команды, задаваемые **общим для всех устройством управления**, причем каждый ПЭ работает с отдельным элементом вектора. ПЭ соединены через **коммутационное устройство** с **многомодульной памятью**.

Исполнение векторной команды включает чтение из памяти элементов векторов, распределение их по процессорам, выполнение заданной операции и засылку результатов обратно в память. Таким образом, производительность системы оказывается равной сумме производительностей всех процессорных элементов.



# Вычислительные системы класса MIMD

MIMD архитектуры различаются в зависимости от того, имеет ли процессор свою собственную локальную память и обращается к другим блокам памяти, используя коммутирующую сеть, или коммутирующая сеть подсоединяет все процессоры к общедоступной памяти.

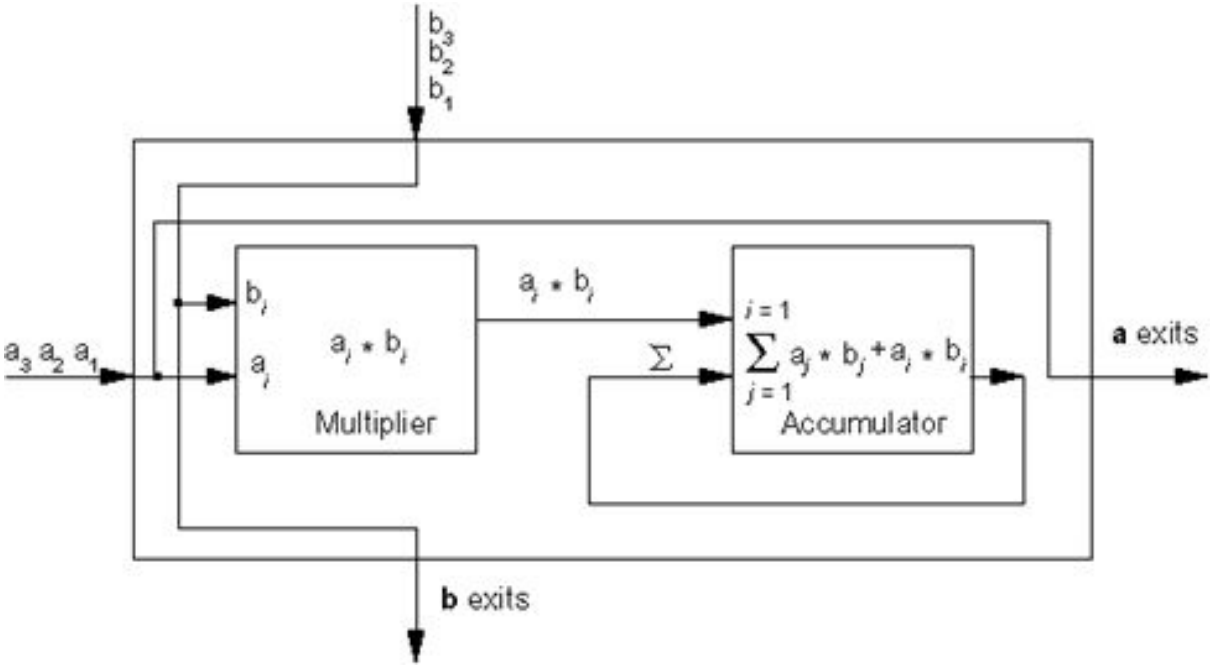
**Систолические вычислительные системы**

Систолические системы являются очень **специализированными** вычислителями и производятся под конкретную задачу. Фактически, **задача построения систолического вычислителя сводится к построению аппаратного конвейера, имеющего достаточно большое время получения результата (т.е. большое количество ступеней), но при этом сравнительно маленькое время между последовательной выдачей результатов, так как значительное количество промежуточных значений обрабатывается на разных ступенях конвейера.**

***Однородные вычислительные структуры*** или среды (**ОВС**), как правило, относятся к типу MIMD и представляют собой **регулярную решетку** из однотипных *процессорных элементов* (ПЭ). Каждый ПЭ обладает алгоритмически полным набором операций, а также операциями обмена или взаимодействия с другими ПЭ. ОВС реализуется на основе микропроцессоров.

# Базовые принципы построения систолических архитектур

1. Систола представляет собой сеть связанных вычислительных ячеек, обычно простых;
2. Каждая ячейка содержит в себе буферный входной регистр для данных и вычислитель, оперирующий с содержимым этого регистра. Выход вычислителя может подаваться на входы других ячеек;
3. Операции в систоле производятся по типу конвейерной обработки;
4. Вычисления в систоле регулируются с помощью общего тактового сигнала;



# Базовые принципы построения систолических архитектур

Основные характеристики систолической ВА:

- однородность процессорного поля,
- регулярность (постоянство) межпроцессорных соединений,
- синхронностью функционирования процессорных элементов.

**В каждый момент времени выполняются одновременные одинаковые операции или одинаковые вычислительные модули.**

Такими модулями могут быть:

- **модули обработки данных и вычислений**
- **модули, отвечающие за внешнюю коммуникацию**.

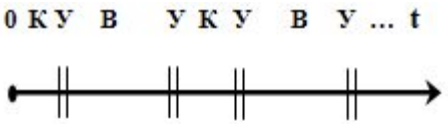
Каждый из двух типов этих модулей выполняется в свою фазу обработки.

***Фазы обработки систолических ВА:***

**К**: внешняя коммуникация между ПЭ;

**В**: вычисления в ПЭ;

**У**: управление вычислениями и коммуникацией (очень короткая).



# Фазы обработки систолических ВА

***Фаза коммуникации***. В этот интервал времени во всей процессорной сети одновременно происходит обмен данными между ПЭ. Интервал должен по длительности соответствовать наиболее долгой операции коммуникации в сети.

***Фаза вычисления***. Осуществляет вычисления и обработку информации. Длительность данной фазы должна соответствовать наиболее продолжительному вычислительному модулю.

***Фаза управления***. Осуществляет выполнение операций по старту и окончанию работ процессорного поля (соответствуют началу и концу каждой вычислительной операции). Останов обработки процессов в любой момент времени до получения результата.

**Применение систолических ВА**:

- ускорители, встроенные в ПК и реализующие конкретные вычислительные алгоритмы (матричные операции, решение систем линейных алгебраических уравнений, распознавание образов, сортировка и др. ). В этом случае процессорная плата используется в качестве сопроцессора. Время вычислений сокращается на 1 – 3 порядка.
- систолические процессоры, встроенные в технические системы, которые используются для цифровой обработки в реальном масштабе времени. Например, алгоритм цифровой фильтрации и др.

# *Массивно-параллельные компьютеры (MPP) с распределенной памятью*

Система состоит из однородных вычислительных узлов, включающих:

- один или несколько центральных процессоров (обычно RISC),
- локальную память (прямой доступ к памяти других узлов невозможен),
- коммуникационный процессор или сетевой адаптер
- иногда - жесткие диски и/или другие устройства В/В

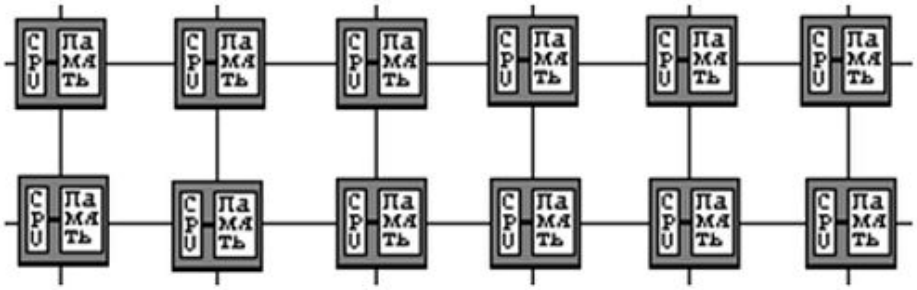
К системе могут быть добавлены специальные узлы ввода-вывода и управляющие узлы. Узлы связаны через некоторую коммуникационную среду (высокоскоростная сеть, коммутатор и т.п.)

Используются два варианта работы операционной системы (ОС) на машинах MPP-архитектуры:

- полноценная операционная система (ОС) работает только на управляющей машине, на каждом отдельном модуле функционирует сильно урезанный вариант ОС, обеспечивающий работу только расположенной в нем ветви параллельного приложения.
- на каждом модуле работает полноценная UNIX-подобная ОС, устанавливаемая отдельно.

# Массивно-параллельные компьютеры (MPP) с распределенной памятью

Общее число процессоров в реальных системах достигает нескольких тысяч.



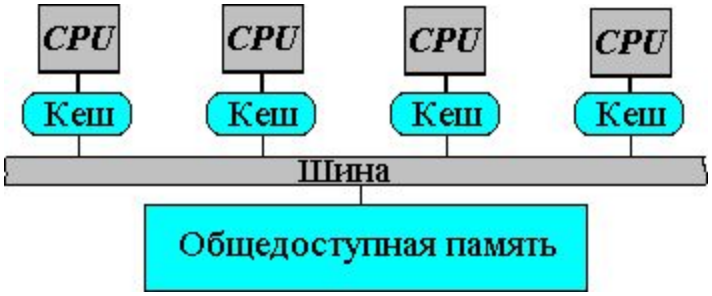

**Преимущество** систем с распределенной памятью - **хорошая масштабируемость**: в машинах этого класса каждый процессор имеет доступ только к своей локальной памяти, в связи с чем не возникает необходимости в потактовой синхронизации процессоров.

**Недостатки**:

- отсутствие общей памяти заметно снижает скорость межпроцессорного обмена. Требуется специальная техника программирования для реализации обмена сообщениями между процессорами;
- каждый процессор может использовать только ограниченный объем локального банка памяти;
- вследствие указанных недостатков требуются значительные усилия для максимального использования системных ресурсов. Это определяет высокую цену программного обеспечения для массивно-параллельных систем с раздельной памятью.

# Компьютеры с общей (разделяемой) памятью (SMP)

SMP (symmetric multiprocessing) – ***симметричная многопроцессорная архитектура***. Главная особенность систем с архитектурой SMP - наличие ***общей физической памяти***, разделяемой всеми процессорами.




Память служит для передачи сообщений между процессорами, при этом все вычислительные устройства при обращении к ней имеют равные права и одну и ту же адресацию для всех ячеек памяти. Поэтому SMP-архитектура называется **симметричной**.

SMP-система строится на основе высокоскоростной системной шины, к слотам которой подключаются функциональные блоки типов: процессоры (ЦП), подсистема ввода/вывода (I/O) и т. п. Вся система работает под управлением единой ОС. ОС автоматически (в процессе работы) распределяет процессы по процессорам, но иногда возможна и явная привязка.

# Компьютеры с общей (разделяемой) памятью (SMP)

**Основные преимущества SMP-систем**:

- простота и универсальность для программирования: обычно используется модель параллельных ветвей, когда все процессоры работают независимо друг от друга. Однако можно реализовать и модели, использующие межпроцессорный обмен. Использование общей памяти увеличивает скорость такого обмена, пользователь также имеет доступ сразу ко всему объему памяти.
- простота эксплуатации. Как правило, SMP-системы используют систему кондиционирования, основанную на воздушном охлаждении, что облегчает их техническое обслуживание;
- относительно невысокая цена.

Недостатки:

**системы с *общей памятью* плохо масштабируются**.

Этот существенный недостаток SMP-систем не позволяет считать их по-настоящему перспективными. Причиной плохой *масштабируемости* является то, что в данный момент шина способна обрабатывать только одну транзакцию. при одновременном обращении нескольких процессоров к одним и тем же областям *общей физической памяти* возникают **проблемы разрешения конфликтов**.

Все процессоры совместно обращаются к общей памяти, обычно через шину или иерархию шин. В идеализированной модели, любой процессор может обращаться к любой ячейке памяти за одно и то же время. На практике масштабируемость этой архитектуры обычно приводит к некоторой форме иерархии памяти. Чтобы сгладить разрыв в скорости работы процессора и основной памяти, каждый процессор снабжается скоростной буферной памятью (кэш-памятью), работающей со скоростью процессора. В связи с этим в многопроцессорных системах, построенных на базе таких микропроцессоров, нарушается принцип равноправного доступа к любой точке памяти и возникает новая проблема - **проблема кэш-когерентности**.

# Компьютеры с общей (разделяемой) памятью (SMP)

Для обеспечения ***когерентности*** кэш-памяти существует несколько возможностей:

- использовать механизм отслеживания шинных запросов, в котором кэши отслеживают переменные, передаваемые к любому из центральных процессоров, и при необходимости модифицируют собственные копии таких переменных;
- выделять специальную часть памяти, отвечающую за отслеживание достоверности всех используемых копий переменных.

**Основное преимущество** систем SMP - **относительная простота программирования**. Т.к. все процессоры имеют одинаково быстрый доступ к ОП, вопрос о том, какой процессор какие вычисления будет выполнять, не столь принципиален, и значительная часть вычислительных алгоритмов, разработанных для однопроцессорных компьютеров, может ускоренно выполняться в мультипроцессорных системах с использованием распараллеливающих и «векторизирующих» компиляторов. Системы SMP - это наиболее распространенный сейчас тип параллельных ВС. В реальных системах можно задействовать не более 32 процессоров.

**Системы MPP позволяют создавать системы с наиболее высокой производительностью. Узлами таких систем часто являются системы SMP.**

# *Компьютеры с виртуальной общей (разделяемой) памятью (NUMA - системы)*

Архитектура **NUMA (nonuniform memory access)** объединяет достоинства классов систем SMP (относительная простота разработки программ) и систем MPP (хорошая масштабируемость - возможность наращивания числа процессорных узлов в системе).

Главная ее особенность – ***неоднородный доступ к памяти***. Суть в особой организации памяти. Память физически распределена по различным частям системы, но логически она является ***общей***, так что пользователь видит ***единое адресное пространство***. Каждый из однородных модулей состоит из небольшого числа процессоров и блока памяти. Модули объединены с помощью ***высокоскоростного коммутатора***. Поддерживается *единое адресное пространство*, аппаратно поддерживается доступ к удаленной памяти, т.е. к памяти других модулей. **При этом доступ к *локальной памяти* осуществляется в несколько раз быстрее, чем к удаленной**. По существу, архитектура NUMA является MPP (*массивно-параллельной*) архитектурой, где в качестве отдельных вычислительных элементов берутся SMP (симметричная многопроцессорная архитектура) узлы. Доступ к памяти и обмен данными внутри одного SMP-узла осуществляется через ***локальную память* узла** и происходит очень быстро, а к процессорам другого SMP-узла тоже есть доступ, но более медленный и через более сложную систему адресации.

# Кластерные системы

***Кластер*** – это два или более компьютеров (узлов), объединяемые при помощи ***сетевых технологий*** на базе **шинной архитектуры** или **коммутатора** и являющиеся для пользователя единым информационно-вычислительным ресурсом.

В качестве узлов *кластера* могут быть выбраны серверы, рабочие станции и даже обычные персональные компьютеры. Узел характеризуется тем, что на нем работает единственная копия операционной системы. Преимущество кластеризации для повышения работоспособности становится очевидным в случае сбоя какого-либо узла: при этом другой узел *кластера* может взять на себя нагрузку неисправного узла, и пользователи не заметят прерывания в доступе. Возможности *масштабируемости кластеров* позволяют многократно увеличивать производительность приложений для большего числа пользователей технологий на базе шинной архитектуры или коммутатора. Такие суперкомпьютерные системы являются самыми дешевыми, поскольку собираются на базе ***стандартных комплектующих элементов***, процессоров, коммутаторов, дисков и внешних устройств.

# Кластерные системы

**Типы кластеров**

**Класс I**. Класс машин строится целиком из стандартных деталей, которые продают многие поставщики компьютерных компонентов (*низкие цены, простое обслуживание, аппаратные компоненты доступны из различных источников*).

**Класс II**. Система имеет эксклюзивные или не слишком широко распространенные детали. Так можно достичь очень хорошей производительности, но при более высокой стоимости.

Способ соединения процессоров друг с другом в большей степени определяет ее производительность, чем тип используемых в ней процессоров. **Критический параметр - расстояние между процессорами** (определяет величину производительности такой системы). Поэтому иногда целесообразнее создать систему из большего числа дешевых компьютеров, чем из меньшего числа дорогих.

В кластерах используются операционные системы, стандартные для рабочих станций, (например, *свободно распространяемые* **Linux**, **FreeBSD**), вместе со специальными средствами поддержки **параллельного программирования и *балансировки нагрузки*.** Для соединения компьютеров в кластер наиболее широко в данное время используется технология **Fast Ethernet** (простота ее использования и низкая стоимость коммуникационного оборудования).

# Закон Амдала и его следствия

**Увеличение количества процессоров не приводит к пропорциональному возрастанию производительности.**

Причины

1.Отсутствие максимального параллелизма в алгоритме и/или несбалансированность нагрузки процессоров.

2. Обмены, конфликты памяти и время синхронизации.

Предположим, что в вашей программе доля операций, которые нужно выполнять последовательно, равна f, где 0 <= f <= 1 .

***Ускорением*** параллельного алгоритма называется отношение:

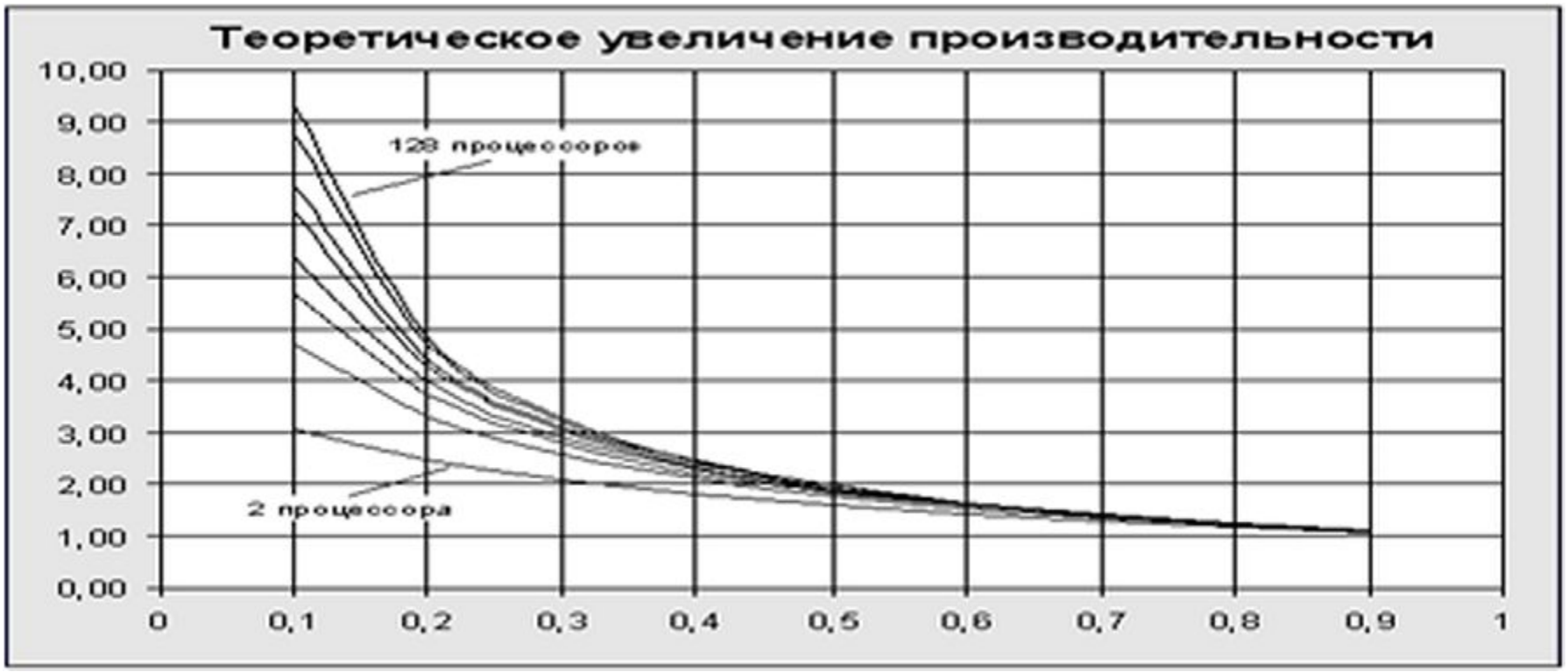
$$S_p = \frac{\text{время выполнения алгоритма на одном процессоре}}{\text{время выполнения алгоритма в системе из P процессоров}}$$

Ускорение позволяет сравнивать поведение данного алгоритма для одного и ***p*** процессоров.

Чтобы оценить, какое ускорение S может быть получено на компьютере из **p** процессоров при данном значении f, можно воспользоваться законом Амдала:

$$S \le \frac{1}{1+(1-f)/p}$$

# Закон Амдала и его следствия



# Нейровычислительная архитектура

Для повышения производительности ЭВМ необходимо перейти от принципов фон-Неймана к параллельной обработке информации. Тем не менее, параллельные компьютеры пока не получили распространения по нескольким причинам

Одним из вариантов реализации классов архитектур вычислительных систем является нейрокомпьютер.

**Нейрокомпьютер** - это вычислительная система с MIMD архитектурой, которая представляет собой совокупность очень простых однотипных **процессорных элементов** (**нейронов**), объединенных множественными связями.

Основные преимущества нейрокомпьютеров связаны с массовым параллелизмом обработки, который обуславливает высокое быстродействие при низких требованиях к параметрам элементарных узлов. Устойчивые и надёжные нейросистемы могут создаваться из низконадёжных элементов, имеющих большой разброс параметров

Любой нейрокомпьютер по своему строению представляет нейронную сеть (нейросеть). **Нейронная сеть** - это сеть с конечным числом слоёв, состоящих из однотипных элементов - аналогов нейронов с различными типами связи между слоями.

Элементарным строительным элементом нейронной сети (НС) является нейрон, который осуществляет **взвешенное суммирование** поступающих на его вход **сигналов**. **Результат** такого суммирования **образует промежуточный выходной сигнал**, **который преобразуется активационной функцией в выходной сигнал нейрона**.

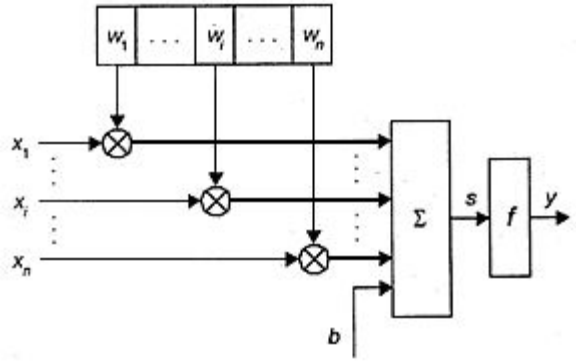
# Решаемые задачи

**Задачи, успешно решаемые НС на данном этапе их развития:**

- формирование моделей и различных нелинейных и трудно описываемых математически систем, прогнозирование развития этих систем во времени:
- системы управления и регулирования с предсказанием; управление роботами, другими сложными устройствами
- разнообразные конечные автоматы: системы массового обслуживания и коммутации, телекоммуникационные системы;
- распознавание зрительных, слуховых образов;
- ассоциативный поиск информации и создание ассоциативных моделей; синтез речи; формирование естественного языка;
- принятие решений и диагностика в областях, где отсутствуют четкие математические модели: в медицине, криминалистике, финансовой сфере;

# Структура и свойства искусственного нейрона

Нейрон состоит из элементов трех типов: **умножителей (синапсов), сумматора и нелинейного преобразователя.** Синапсы осуществляют связь между нейронами, умножают входной сигнал на число, характеризующее силу связи, (вес синапса). Сумматор выполняет сложение сигналов, поступающих по синаптическим связям от других нейронов, и внешних входных сигналов. Нелинейный преобразователь реализует нелинейную функцию одного аргумента - выхода сумматора. Эта функция называется **функцией активации** или передаточной функцией нейрона.



# Структура и свойства искусственного нейрона

Нейрон реализует скалярную функцию векторного аргумента

Математическая модель нейрона:

$$s = \sum_{i=1}^{n} w_i x_i + b$$

$$y = f(s)$$

где $w_i$, - вес синапса, **i** = 1...n; **b** - значение смещения; **s** - результат суммирования; $x_i$ - компонент входного вектора (входной сигнал), **i** = 1...n;

**y** - выходной сигнал нейрона; **n** - число входов нейрона; **f** - нелинейное преобразование (функция активации).

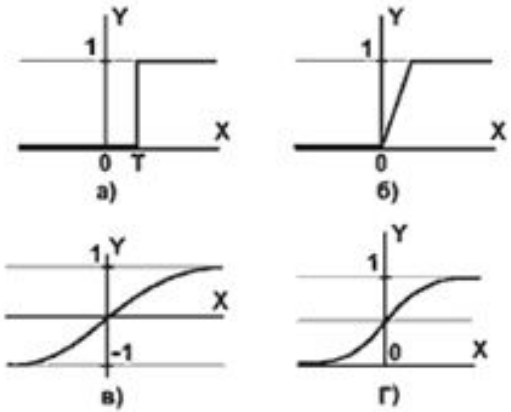
В общем случае входной сигнал, весовые коэффициенты и смещение могут принимать действительные значения. Выход (**y**) определяется видом функции активации и может быть как действительным, так и целым.

Синаптические связи с положительными весами называют **возбуждающими**, с отрицательными весами – **тормозящими.**

Описанный вычислительный элемент можно считать упрощенной математической моделью биологических нейронов

# Структура и свойства искусственного нейрона

На входной сигнал (**s**) нелинейный преобразователь отвечает выходным сигналом **f(s)**, который представляет собой выход у нейрона.




Одной из наиболее распространенных является нелинейная функция активации с насыщением (логистическая функция или сигмоид (функция S-образного вида):

$$f(s) = 1 / (1+e^{-as})$$

При уменьшении **a** сигмоид становится более пологим, в пределе при a = 0 вырождаясь в горизонтальную линию на уровне 0,5, при увеличении **a** сигмоид приближается к виду функции единичного скачка с порогом 0. Из выражения для сигмоида очевидно, что выходное значение нейрона лежит в диапазоне (0, 1). Одно из ценных свойств сигмоидальной функции - простое выражение для ее производной. Кроме того, она обладает свойством усиливать слабые сигналы лучше, чем большие, и предотвращает насыщение от больших сигналов, так как они соответствуют областям аргументов, где сигмоид имеет пологий наклон.

# Синтез нейронных сетей

В зависимости от функций, выполняемых нейронами в сети, можно выделить три типа нейронов:

- **входные нейроны**, на которые подается вектор, кодирующий входное воздействие или образ внешней среды; в них обычно не осуществляется вычислительных процедур, а информация передается с входа на выход путем изменения их активации;
- **выходные нейроны**, выходные значения которых представляют выходы нейронной сети; преобразования в них осуществляются по выражениям (1.1) и (1.2);
- **промежуточные нейроны**, составляющие основу нейронных сетей, преобразования в которых выполняются также по выражениям (1.1) и (1.2).

В большинстве нейронных моделей тип нейрона связан с его расположением в сети. Если нейрон имеет только выходные связи, то это входной нейрон, если наоборот - выходной нейрон. В процессе функционирования сети осуществляется преобразование входного вектора в выходной, некоторая переработка информации.

Известные нейронные сети можно разделить по типам структур нейронов на *гомогенные* (однородные) и *гетерогенные*. Гомогенные сети состоят из нейронов одного типа с единой функцией активации, а в гетерогенную сеть входят нейроны с различными функциями активации.

# Выбор количества нейронов и слоев

Количество нейронов и слоев связано:

1) со сложностью задачи;
2) с количеством данных для обучения;
3) с требуемым количеством входов и выходов сети;
4) с имеющимися ресурсами: памятью и быстродействием машины, на которой моделируется сеть;

**Если в сети слишком мало нейронов или слоев:**

1) сеть не обучится и ошибка при работе сети останется большой;
2) на выходе сети не будут передаваться резкие колебания аппроксимируемой функции y(x).

**Если нейронов или слоев слишком много:**

1) быстродействие будет низким, а памяти потребуется много на фон-неймановских ЭВМ;
2) сеть переобучится: выходной вектор будет передавать незначительные и несущественные детали в изучаемой зависимости y(x), например, шум или ошибочные данные;
3) зависимость выхода от входа окажется резко нелинейной: выходной вектор будет существенно и непредсказуемо меняться при малом изменении входного вектора x;
4) сеть будет неспособна к обобщению: в области, где нет или мало известных точек функции y(x) выходной вектор будет случаен и непредсказуем, не будет адекватен решамой задаче

# Подготовка входных и выходных данных

Данные, подаваемые на вход сети и снимаемые с выхода, должны быть правильно подготовлены.

Один из распространенных способов - **масштабирование:**

$$x = (x' - \mathbf{m}) \cdot c$$

где x - исходный вектор, $x'$ - масштабированный. Вектор **m** - усредненное значение совокупности входных данных, c - масштабный коэффициент.

Масштабирование желательно, чтобы привести данные в допустимый диапазон. Если этого не сделать, то возможно несколько проблем:

1) нейроны входного слоя или окажутся в постоянном насыщении (|m| велик, дисперсия входных данных мала) или будут все время заторможены (|m| мал, дисперсия мала);

2) весовые коэффициенты примут очень большие или очень малые значения при обучении (в зависимости от дисперсии), и, как следствие, растянется процесс обучения и снизится точность

# Обучение сети

Процесс функционирования НС зависит от величин синаптических связей, поэтому, задавшись определенной структурой НС, отвечающей какой-либо задаче, разработчик сети должен найти оптимальные значения всех переменных весовых коэффициентов (некоторые синаптические связи могут быть постоянными).

Этот этап называется **обучением НС**. Способность сети решать поставленные перед ней проблемы во время эксплуатации зависит от того, насколько качественно он будет выполнен. На этапе обучения кроме параметра **качества подбора весов** важную роль играет **время обучения**. Как правило, эти два параметра связаны обратной зависимостью и их приходится выбирать на основе компромисса.

Обучение НС может вестись **с учителем** или **без него**.

В первом случае сети предъявляются значения как входных, так и желательных выходных сигналов, и она по некоторому внутреннему алгоритму подстраивает веса своих синаптических связей.

Во втором случае выходы НС формируются самостоятельно, а веса изменяются по алгоритму, учитывающему только входные и производные от них сигналы.

# Обучение сети

Алгоритмы обучения делятся на два больших класса: **детерминистские** и **стохастические**. В первом из них подстройка весов представляет собой жесткую последовательность действий, во втором – она производится на основе действий, подчиняющихся некоторому случайному процессу.

Рассмотрим алгоритм обучения с учителем.

1. Проинициализировать элементы весовой матрицы (обычно небольшими случайными значениями).
2. Подать на входы один из входных векторов, которые сеть должна научиться различать, и вычислить ее выход.
3. Если выход правильный, перейти на шаг 4.

Иначе вычислить разницу между идеальным и полученным значениями выхода:

Модифицировать веса в соответствии с определенной формулой

4. Цикл с шага 2, пока сеть не перестанет ошибаться.

На втором шаге на разных итерациях поочередно в случайном порядке предъявляются все возможные входные вектора. К сожалению, нельзя заранее определить число итераций, которые потребуется выполнить, а в некоторых случаях и гарантировать полный успех.