Общая характеристика многопроцессорных вычислительных систем

Гергель В.П., профессор,
д.т.н.
Нижегородский университет

- Классификация многопроцессорных вычислительных систем
  - Мультипроцессоры – системы с общей памятью
  - Мультикомпьютеры – системы с распределенной памятью
- Типовые схемы коммуникации процессоров
- Системные платформы для построения кластеров

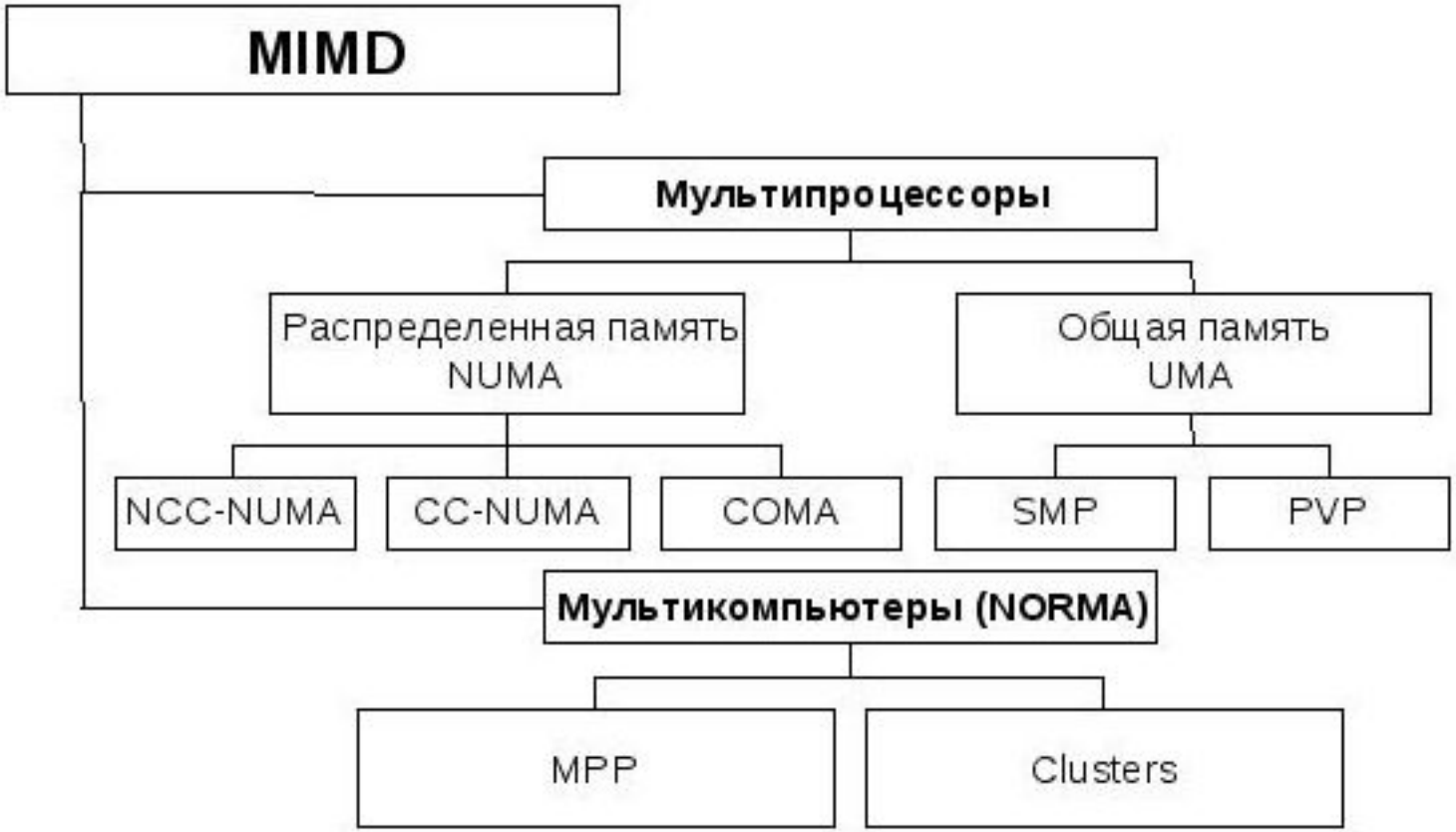
Классификация вычислительных систем…

- **Систематика Флинна** (Flynn)
  - Классификация по способам взаимодействия последовательностей (*потоков*) выполняемых команд и обрабатываемых данных:
    - **SISD** (Single Instruction, Single Data)
    - **SIMD** (Single Instruction, Multiple Data)
    - **MISD** (Multiple Instruction, Single Data)
    - **MIMD** (Multiple Instruction, Multiple Data)

*Практически все виды параллельных систем, несмотря на их существенную разнородность, относятся к одной группе **MIMD***

- **Детализация систематики Флинна…**
  - Дальнейшее разделение типов многопроцессорных систем основывается на используемых способах организации оперативной памяти,
  - Позволяет различать два важных типа многопроцессорных систем:
    - ***multiprocessors*** (***мультипроцессоры*** или системы с общей разделяемой памятью),
    - ***multicomputers*** (***мультикомпьютеры*** или системы с распределенной памятью).

Детализация систематики Флинна…

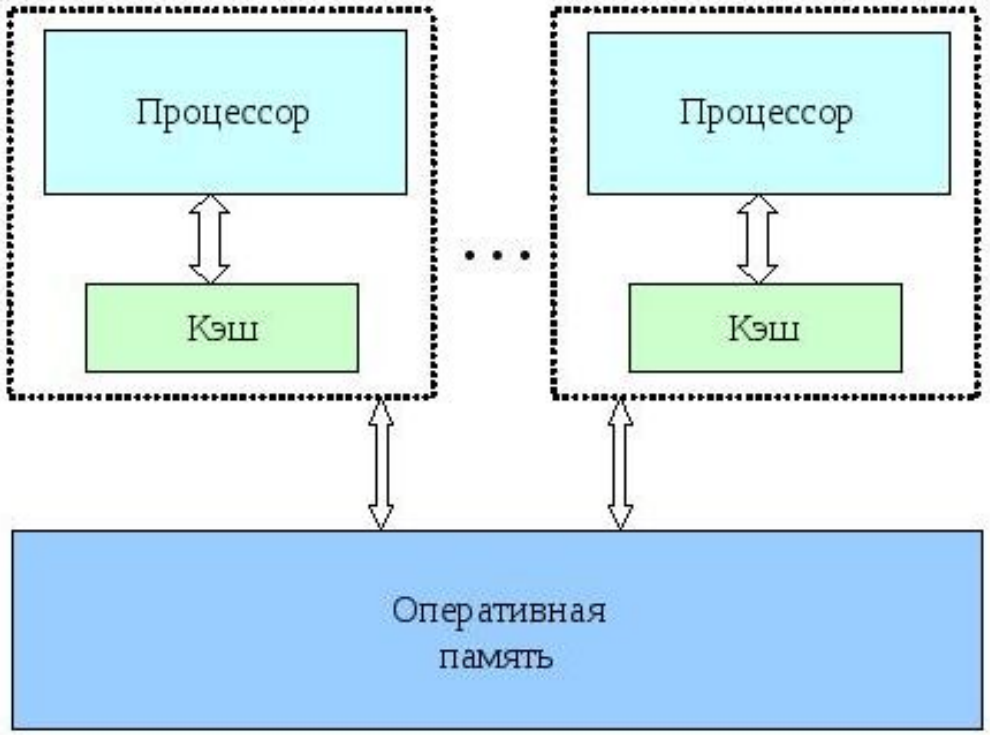


Мультипроцессоры с использованием единой *общей памяти* (*shared memory*)…

- Обеспечивается *однородный доступ к памяти* (*uniform memory access or UMA*),
- Являются основой для построения:
  - *векторных параллельных процессоров* (*parallel vector processor or PVP*). Примеры: Cray T90,
  - симметричных *мультипроцессоров* (*symmetric multiprocessor or SMP*). Примеры: IBM eServer, Sun StarFire, HP Superdome, SGI Origin.

- **Мультипроцессоры** с использованием единой *общей памяти* (*shared memory*)…
  - Обеспечивается *однородный доступ к памяти* (*uniform memory access or UMA*),
  - Являются основой для построения:
    - *векторных параллельных процессоров* (*parallel vector processor or PVP*). Примеры: Cray T90,
    - симметричных *мультипроцессоров* (*symmetric multiprocessor or SMP*). Примеры: IBM eServer, Sun StarFire, HP Superdome, SGI Origin.

- **Мультипроцессоры** с использованием единой *общей памяти…*

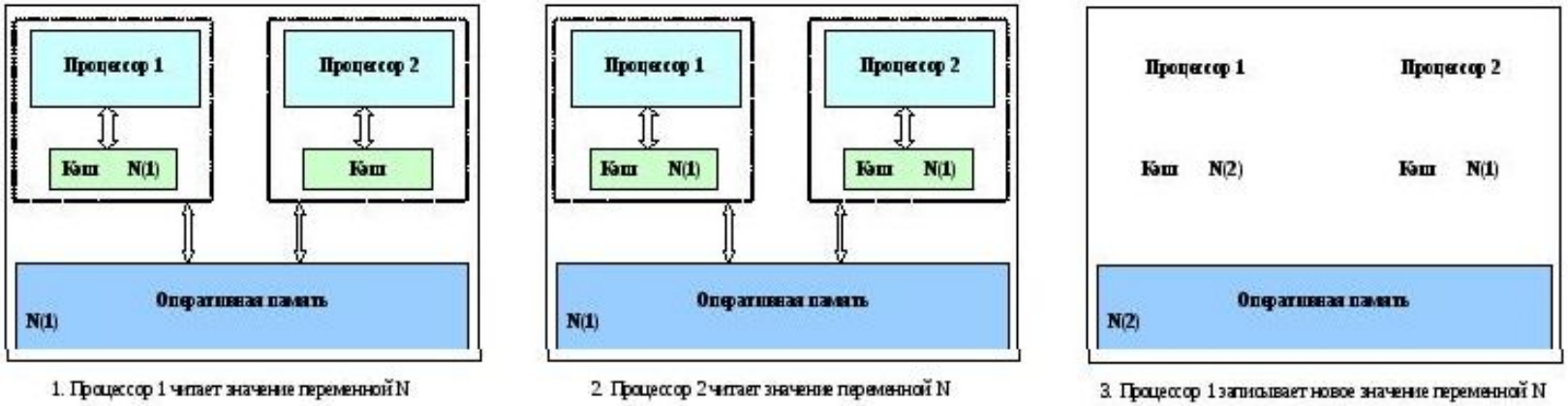


- **Мультипроцессоры** с использованием единой *общей памяти…*

Проблемы:

- Доступ с разных процессоров к общим данным и обеспечение, в этой связи, *однозначности (когерентности) содержимого разных кэшей* (*cache coherence problem*),
- Необходимость *синхронизации взаимодействия* одновременно выполняемых потоков команд

Проблема: *Обеспечение однозначности (когерентности) содержимого разных кэшей (cache coherence problem)*



1. Процессор 1 читает значение переменной N

2. Процессор 2 читает значение переменной N

3. Процессор 1 записывает новое значение переменной N

При изменении данных необходимо проверять наличие "старых" значений в кэш-памяти всех процессоров (обеспечивается на аппаратном уровне, но становится сложным при большом количестве процессоров)

- **Мультипроцессоры** с использованием единой *общей памяти…*

Проблема: *Необходимость синхронизации взаимодействия одновременно выполняемых потоков команд…*

Пример: Пусть процессоры выполняют последовательность команд

```
N = N + 1
Печать N
```

над общей переменной *N* (в скобках указывается значение этой переменной)

Вариант исполнения 1

Время	Процессор 1	Процессор 2
1	Чтение N (1)	
2		Чтение N (1)
3		Прибавление 1 (2)
4	Прибавление 1 (2)	
5	Запись N (2)	
6	Печать N (2)	
7		Запись N (2)
8		Печать N (2)

Вариант исполнения 2

Время	Процессор 1	Процессор 2
1	Чтение N (1)	
2	Прибавление 1 (2)	
3	Запись N (2)	
4	Печать N (2)	
5		Чтение N (2)
6		Прибавление 1 (3)
7		Запись N (3)
8		Печать N (3)

Временная последовательность команд может быть различной – необходима синхронизация при использовании общих переменных!

Проблема*: Необходимость синхронизации взаимодействия одновременно выполняемых потоков команд…*

Рассмотренный пример может рассматриваться как проявление общей *проблемы использования разделяемых ресурсов* (общих данных, файлов, устройств и т.п.)

Для организации разделения ресурсов между несколькими потоками команд необходимо иметь возможность:

- *определения доступности* запрашиваемых ресурсов (ресурс свободен и может быть выделен для использования, ресурс уже занят одним из потоков и не может использоваться дополнительно каким-либо другим потоком);

- *выделения свободного ресурса* одному из процессов, запросивших ресурс для использования;

- *приостановки (блокировки) потоков*, выдавших запросы на ресурсы, занятые другими потоками.

- **Мультипроцессоры** с использованием единой *общей памяти…*

Проблема: *Необходимость синхронизации взаимодействия одновременно выполняемых потоков команд…*

- Рассмотренный пример может рассматриваться как проявление общей *проблемы использования разделяемых ресурсов* (общих данных, файлов, устройств и т.п.)
- Для **организации разделения ресурсов** между несколькими потоками команд необходимо иметь возможность:

  - *определения доступности* запрашиваемых ресурсов (ресурс свободен и может быть выделен для использования, ресурс уже занят одним из потоков и не может использоваться дополнительно каким-либо другим потоком);

  - *выделения свободного ресурса* одному из процессов, запросивших ресурс для использования;

  - *приостановки (блокировки) потоков*, выдавших запросы на ресурсы, занятые другими потоками.

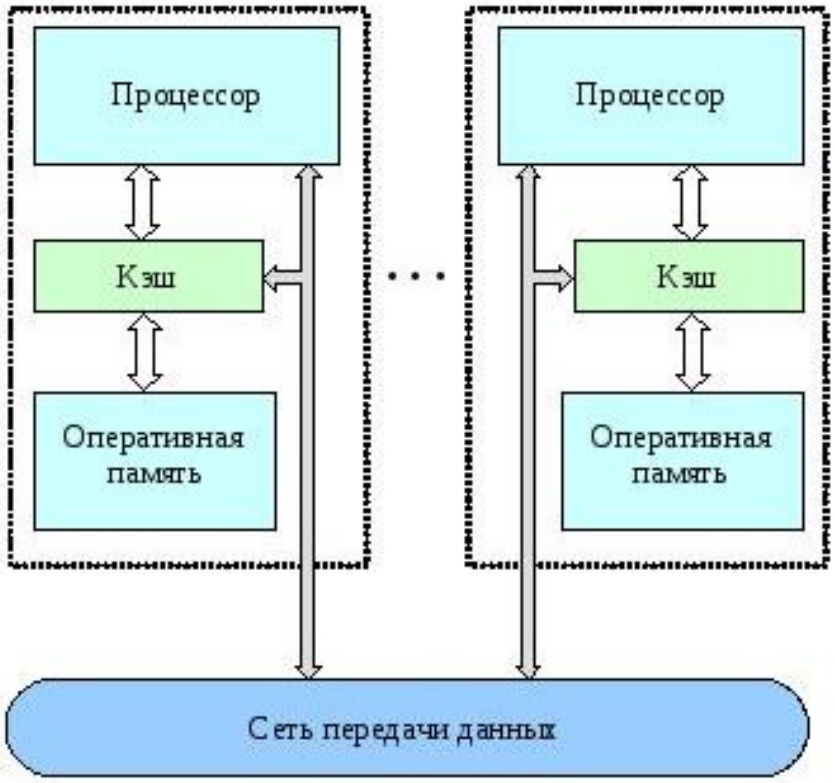
Проблема: *Необходимость синхронизации взаимодействия одновременно выполняемых потоков команд*

- Доступ к общей переменной в рассмотренном примере в самом общем виде должен быть организован следующим образом:

```
<Получить доступ>
  N = N + 1
  Печать N
<Завершить доступ>
```

- **Мультипроцессоры** с использованием физически распределенной памяти (*distributed shared memory* or *DSM*):
  - *Неоднородный доступ к памяти* (*non-uniform memory access or NUMA*),
  - Среди систем такого типа выделяют:
    - *cache-only memory architecture or COMA* (системы KSR-1 и DDM),
    - *cache-coherent NUMA or CC-NUMA* (системы SGI Origin 2000, Sun HPC 10000, IBM/Sequent NUMA-Q 2000),
    - *non-cache coherent NUMA or NCC-NUMA* (система Cray T3E).

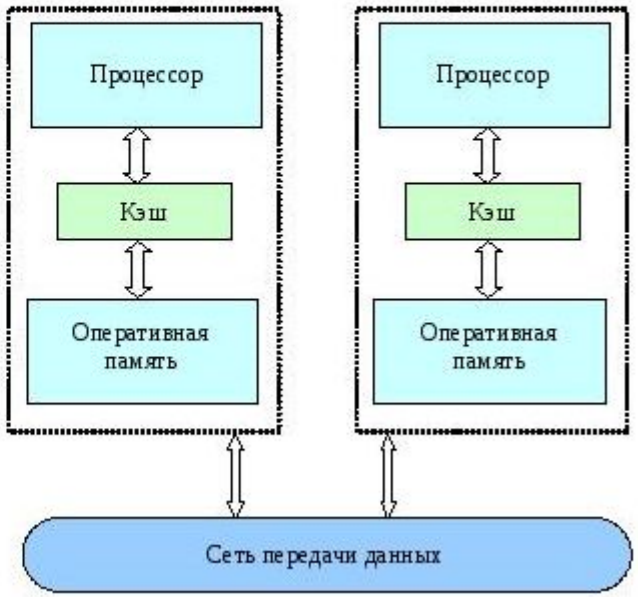
- **Мультипроцессоры** с использованием физически распределенной памяти…



- **Мультипроцессоры** с использованием физически распределенной памяти:
  - Упрощаются проблемы создания мультипроцессоров (известны примеры систем с несколькими тысячами процессоров),
  - Возникают проблемы эффективного использования распределенной памяти (время доступа к локальной и удаленной памяти может различаться на несколько порядков).

Классификация вычислительных систем…

- **Мультикомпьютеры…**
  - Не обеспечивают общий доступ ко всей имеющейся в системах памяти (*no-remote memory access or NORMA*),
  - Каждый процессор системы может использовать только свою локальную память



Мультикомпьютеры…

- Для доступа к данным, располагаемых на других процессорах, необходимо явно выполнить *операции передачи сообщений* (*message passing operations*),
- Основные операции передачи данных:
  - Отправить сообщение (*send*),
  - Получить сообщение (*receive*)

Пример:

Процессор 1
<Отправить сообщение>
<Продолжение вычислений>

Процессор 2
<Получить сообщение>
<Продолжение вычислений с использованием данных полученного сообщения>

- **Мультикомпьютеры**

  Данный подход используется при построении двух важных типов многопроцессорных вычислительных систем:

  - *массивно-параллельных систем* (*massively parallel processor or MPP*), например: IBM RS/6000 SP2, Intel PARAGON, ASCI Red, транспьютерные системы Parsytec,
  - *кластеров* (*clusters*), например: AC3 Velocity и NCSA NT Supercluster.

- **Мультикомпьютеры. Кластеры…**

Кластер *- множество отдельных компьютеров, объединенных в сеть, для которых при помощи специальных аппаратно-программных средств обеспечивается возможность унифицированного управления (single system image), надежного функционирования (availability) и эффективного использования (performance)*

- **Мультикомпьютеры. Кластеры…**
- **Преимущества:**
  - Могут быть образованы на базе уже существующих у потребителей отдельных компьютеров, либо же сконструированы из типовых компьютерных элементов;
  - Повышение вычислительной мощности отдельных процессоров позволяет строить кластеры из сравнительно небольшого количества отдельных компьютеров (*lowly parallel processing*),
  - Для параллельного выполнения в алгоритмах достаточно выделять только крупные независимые части расчетов (*coarse granularity*).

- **Мультикомпьютеры. Кластеры…**

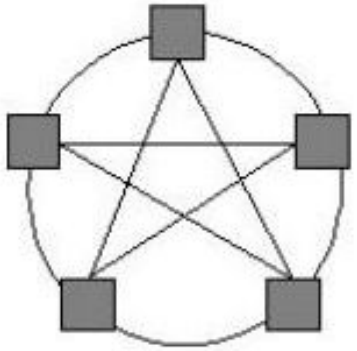
  Преимущества:
  - Могут быть образованы на базе уже существующих у потребителей отдельных компьютеров, либо же сконструированы из типовых компьютерных элементов;
  - Повышение вычислительной мощности отдельных процессоров позволяет строить кластеры из сравнительно небольшого количества отдельных компьютеров (*lowly parallel processing*),
  - Для параллельного выполнения в алгоритмах достаточно выделять только крупные независимые части расчетов (*coarse granularity*).

- **Мультикомпьютеры. Кластеры**

  Недостатки:
  - Организация взаимодействия вычислительных узлов кластера при помощи передачи сообщений обычно приводит к значительным временным задержкам,
  - Дополнительные ограничения на тип разрабатываемых параллельных алгоритмов и программ (*низкая интенсивность потоков передачи данных*)

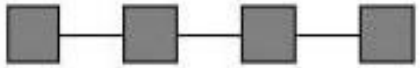
Топология сети передачи данных…

- **полный граф** (*completely-connected graph* or *clique*) – система, в которой между любой парой процессоров существует прямая линия связи,



□□□□□ □□□ (*completely-connected graph* or *clique*)

- **линейка** (*linear array* or *farm*) – система, в которой все процессоры перенумерованы по порядку и каждый процессор, кроме первого и последнего, имеет линии связи только с двумя соседними,

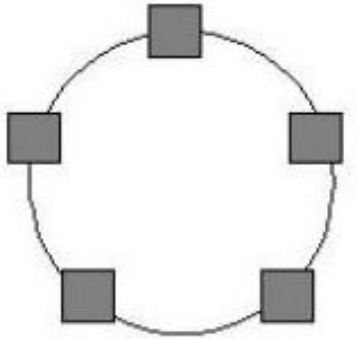


□ □□□□□□ *(linear array or farm)*

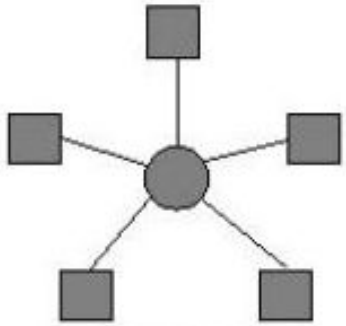
первого и последнего, имеет линии связи только с двумя соседними,

□ □□□□□□ *(linear array or farm)*

- **Топология сети передачи данных…**
  - **кольцо** (*ring*) – данная топология получается из линейки процессоров соединением первого и последнего процессоров линейки,
  - **звезда** (*star*) – система, в которой все процессоры имеют линии связи с некоторым управляющим процессором,



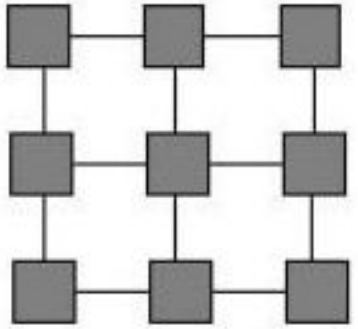
□□□□□□ (*ring*)



□□□□□□ (*star*)

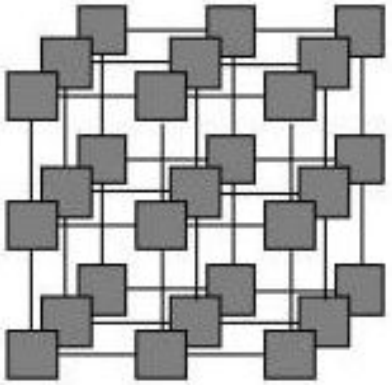
Топология сети передачи данных…

– **решетка** (*mesh*) – система, в которой граф линий связи образует прямоугольную сетку,



□□□ □□□□ (*mesh*)

– **гиперкуб** (*hypercube*) – данная топология представляет частный случай структуры решетки, когда по каждой размерности сетки имеется только два процессора.



- **Топология сети вычислительных кластеров**

  Для построения кластерной системы во многих случаях используют *коммутатор* (*switch*), через который процессоры кластера соединяются между собой.

  Одновременность выполнения нескольких коммуникационных операций является ограниченной.

В любой момент времени каждый процессор может принимать участие только в одной операции приема - передачи данных

Характеристики топологии сети…

- ***диаметр*** – максимальное расстояние между двумя процессорами сети; характеризует максимально-необходимое время для передачи данных между процессорами,
- ***связность*** (*connectivity*) – минимальное количество дуг, которое надо удалить для разделения сети передачи данных на две несвязные области,
- ***ширина бинарного деления*** (*bisection width*) – минимальное количество дуг, которое надо удалить для разделения сети передачи данных на две несвязные области одинакового размера,
- ***стоимость*** – общее количество линий передачи данных в многопроцессорной вычислительной системе.

Характеристика типовых схем коммуникации

Характеристики топологии сети

Топология	Диаметр	Ширина бисекции	Связность	Стоимость
Полный граф	1	$p^2/4$	(p-1)	p(p-1)/2
Звезда	2	1	1	(p-1)
Линейка	p-1	1	1	(p-1)
Кольцо	$\lfloor p/2 \rfloor$	2	2	p
Гиперкуб	$\log_2 p$	p/2	$\log_2 p$	$p \log_2 p/2$
Решетка (N=2)	$2\lfloor \sqrt{p}/2 \rfloor$	$2\sqrt{p}$	4	2p

Характеристика системных платформ для построения кластеров…

- В качестве системной платформы для построения кластеров используют обе наиболее распространенные в настоящий момент операционные системы Unix/Linux и Microsoft Windows.
- Далее подробно будет рассмотрено решение на основе ОС семейства Microsoft Windows; характеристика подхода на базе ОС Unix может быть получена, например, в

Sterling, T. (Ed.) Beowulf Cluster Computing with Linux. - Cambridge, MA: The MIT Press, 2002.

Microsoft Compute Cluster Server 2003…

- Интегрированная платформа для поддержки высокопроизводительных вычислений на кластерных системах
- CCS 2003 состоит из операционной системы Microsoft Windows Server 2003 и Microsoft Compute Cluster Pack (CCP) – набора интерфейсов, утилит и инфраструктуры управления
- Вместе с CCP поставляется SDK, содержащий необходимые инструменты разработки программ для CCS, включая собственную реализацию MPI (Microsoft MPI)

CCS, включая собственную реализацию MPI (Microsoft MPI)

Характеристика системных платформ для построения кластеров…

Microsoft Compute Cluster Server 2003…

- В качестве вычислительных узлов кластера могут быть использованы 64-битные процессоры семейства x86 с, как минимум, 512 Мб оперативной памяти и 4 Гб свободного дискового пространства
- На вычислительных узлах кластера должна быть установлена операционная система Microsoft Windows Server 2003 (Standard, Enterprise или Compute Cluster Edition)

Microsoft Compute Cluster Server 2003

- В состав CCP входит удобная система планирования заданий, позволяющая просматривать состояния всех запущенных задач, собирать статистику, назначать запуски программ на определенное время, завершать "зависшие" задачи и пр.
- В состав CCP входит Microsoft MPI – версия реализации стандарта MPI 2 от Argonne National Labs. MS MPI совместима с MPICH 2 и поддерживает полнофункциональный API с более чем 160 функциями
- Microsoft Visual Studio 2005 включает параллельный отладчик, работающий с MS MPI