Что такое Beowulf

(технология организации параллельных вычислений на Linux-кластерах)

Поставщики традиционных коммерческих суперкомпьютеров достаточно быстро улучшают производительность, надежность и простоту использования своих продуктов. Однако у этих компьютеров есть один большой недостаток - **цена**, подчас недоступная для многих образовательных и научно-исследовательских организаций. Однако потребность в вычислительных ресурсах у этих организаций велика.

Возникла идея создавать параллельные вычислительные системы (кластеры) из общедоступных компьютеров на базе Intel и недорогих Ethernet-сетей, устанавливая на эти компьютеры Linux и одну из бесплатно распространяемых коммуникационных библиотек (PVM, а затем MPI). Оказалось, что на многих классах задач и при достаточном числе узлов такие системы дают производительность, сравнимую с суперкомпьютерной.

История проекта Beowulf

Проект возник в научно-космическом центре NASA - Goddard Space Flight Center (GSFC), точнее в созданном на его основе CESDIS (Center of Excellence in Space Data and Information Sciences).

Изначательно термин "Beowulf" возник как собственное имя Linux-кластера в GSFC. Затем он стал применяться ко всем аналогичным кластерным системам.

Проект Beowulf начался летом 1994 года сборкой в GSFC 16-процессорного кластера (на процессорах 486DX4/100MHz, 16MB памяти и 3 сетевых адаптера на каждом узле, 3 "параллельных" Ethernet-кабеля по 10Mbit).

Как построить Beowulf

Кластер состоит из отдельных машин (*узлов*) и объединяющей их сети (*коммутатора*). Кроме ОС, необходимо установить и настроить сетевые драйверы, компиляторы, ПО поддержки параллельного программирования и распределения вычислительной нагрузки.

1. Узлы кластера. Подходящим выбором в данный момент являются системы на базе процессоров Intel - или однопроцессорные ПК, или SMP-сервера с небольшим числом процессоров (2-4, возможно до 6). По некоторым причинам оптимальным считается построение кластеров на базе **двухпроцессорных** систем, несмотря на то, что в этом случае настройка кластера будет несколько сложнее.

Одну из машин следует выделить в качестве **центральной** (головной) куда следует установить достаточно большой жесткий диск, возможно более мощный процессор и больше памяти, чем на остальные (рабочие) узлы. Имеет смысл обеспечить (защищенную) связь этой машины с внешним миром.

При комплектации **рабочих узлов** (если не планируется их использование в качестве рабочих мест) вполне возможно отказаться от жестких дисков - эти узлы будут загружать ОС через сеть с центральной машины.

Возможна организация кластеров на базе уже существующих сетей рабочих станций, т.е. рабочие станции пользователей могут использоваться в качестве узлов кластера ночью и в выходные дни. Системы такого типа иногда называют COW (Cluster of Workstations).

Количество узлов следует выбирать исходя из необходимых вычислительных ресурсов и доступных финансовых средств. Следует понимать, что при большом числе узлов придется также устанавливать более сложное и дорогое сетевое оборудование.

2. Сеть. Для получения хорошей производительности межпроцессорных обменов используют полнодуплексный Fast Ethernet на 100Mbit/sec. При этом для уменьшения числа коллизий или устанавливают несколько "параллельных" сегментов Ethernet, или соединяют узлы кластера через **коммутатор** (switch).

Более дорогостоящим, но также популярным вариантом являются использование коммутаторов типа Myrinet (1.28Gbit/sec, полный дуплекс). Менее популярными, но также реально используемыми при построении кластеров сетевыми технологиями являются технологии cLAN, SCI и Gigabit Ethernet.

3. Операционная система. Следует установить бесплатно распространяемую ОС Linux. Существует и отлажена техника загрузки Linux через сеть, что очень полезно для бездисковых конфигураций. Необходимо найти и правильно настроить наиболее подходящие к установленным адаптерам драйвера.

4. Компиляторы. Существуют бесплатные компиляторы проекта GNU - GCC/G77, распространяемые вместе с Linux, и коммерческие компиляторы Fortran/C/C++, входящие в пакет PGI Workstation компании Portland Group (PGI).

5. Коммуникационные библиотеки. Наиболее распространенным интерфейсом параллельного программирования в модели передачи сообщений является MPI. Рекомендуемая бесплатная реализация MPI - пакет MPICH, разработанный в Аргоннской Национальной Лаборатории. Для кластеров на базе коммутатора Myrinet разработана система HPVM, куда также входит реализация MPI.

Для эффективной организации параллелизма внутри одной SMP-системы возможны два варианта:

1. Для каждого процессора в SMP-машине порождается отдельный MPI-процесс. MPI-процессы внутри этой системы обмениваются сообщениями через разделяемую память (необходимо настроить MPICH соответствующим образом).
2. На каждой машине запускается только один MPI-процесс. Внутри каждого MPI-процесса производится распараллеливание в модели "общей памяти", например с помощью директив OpenMP.

Кроме MPI, есть и другие библиотеки и системы параллельного программирования, которые могут быть использованы на кластерах. Следует понимать, что использование для программирования нестандартных средств ведет к плохой переносимости параллельных программ.

[1] http://www.beowulf.org

[2] http://parallel.ru/russia/map/data/project15.html

[3] http://www.cacr.caltech.edu/beowulf/tutorial/tutorial.html