Реализации многопоточности

Программирование с использованием POSIX thread library

2006-2007 Иртегов Д.В.

Учебное пособие подготовлено по заказу и при поддержке

ООО «Сан Майкросистемс СПБ»

Как можно реализовать многопоточность

- Пользовательские нити
- Системные нити
- Гибридная реализация (MxN)

Пользовательские нити

- Планировщик в пользовательском адресном пространстве
- + Не требует переделки ядра системы
- + Не требует дополнительных системных ресурсов
- + Легко реализовать
- - Как быть с блокирующимися системными вызовами?
- - Не может использовать несколько процессоров

Примеры пользовательских нитей

- Fibers (волокна) в Win32
  - Не могут исполнять блокирующиеся системные вызовы
  - Поэтому очень редко используются
- Задачи в Minix (когда Minix работает как задача полноценной Unix-системы)

Системные нити

- Для планирования используется системный планировщик
- - Нужна переделка планировщика и процедуры создания процессов
- - Системные процессы считаются дорогим ресурсом (занимают память ядра)
- + Планировщик в ядре так или иначе уже есть
- + Нет проблемы блокирующихся системных вызовов
- + Можно использовать все процессоры в системе

Гибридная реализация (MxN)

- Требует наличия как системного, так и пользовательского планировщика
- Системный планировщик поддерживает M нитей на процесс.
- Пользовательский планировщик поддерживает N нитей (N≥M)
- Пользовательский планировщик распределяет пользовательские нити между системными нитями, подобно тому, как планировщик многозадачной ОС распределяет задания между процессорами

Гибридный планировщик

- + Имеет все преимущества системного планировщика
- + По идее, пользовательская нить должна быть дешевле (для нее не обязательно создается системная нить)
- - Возникает лишняя сущность (пользовательский планировщик)
- - Во многих реальных приложениях M растет и быстро достигает N

Гибридный планировщик в старых версиях Solaris

- Системные нити называются LWP (Light Weight Process – легковесный процесс)
- Системные нити подчинены процессу
- Пользовательских нитей больше, чем системных (во всяком случае, в начале)
- Когда все LWP садятся в блокирующиеся системные вызовы, система посылает сигнал SIGWAITING
- Библиотека ловит SIGWAITING и может создать новый LWP

Гибридный планировщик в Solaris (продолжение)

- Библиотека позволяет привязывать нити к определенному LWP (bound thread)
- Можно управлять количеством LWP (set concurrency)
- В POSIX Thread Library есть API позволяющие добиться того же эффекта (thread scope)
- В Solaris 9 от этого отказались и перешли к системному планировщику (LWP на каждую пользовательскую нить)
- Старые API остались, но их вызовы ничего не делают
- SCO UnixWare, IBM AIX, HP HP/UX по прежнему поддерживают гибридный планировщик

POSIX Threads в Linux

- В Linux в ядре 2.4 есть системные нити (clone(2)). Эти нити имеют собственный PID и собственную запись в таблице процессов
- Linux поддерживает основные функции POSIX Thread API, но есть ряд несовместимостей
- В Linux 2.6 была реализована т.наз. NPTL (Native POSIX Thread Library), более похожая на стандарт POSIX.
- Linux 2.4 и 2.6 используют системные нити (одна системная нить на каждую пользовательскую)

Сборка многопоточных программ

- В большинстве Unix-систем сборка многопоточных программ требует подключения библиотеки libpthread.so
  (cc -lpthread program.c …)
- Также многие компиляторы рекомендуют использовать специальные ключи
  - -mt - Sun Studio C compiler
  - -threads или -pthread - GNU C
    (в зависимости от сборки)
- У старых компиляторов ключ -mt мог подключать другую версию libc
- У современных компиляторов ключ -mt отключает небезопасные оптимизации и определяет препроцессорные символы (_REENTRANT в Solaris)
- Ряд стандартных include-файлов содержат директивы условной компиляции, использующие _REENTRANT, и заменяют некоторые небезопасные конструкции на более приемлемые для многопоточной программы

Еще о сборке (Solaris 10)

- В Solaris 10 библиотека libc.so содержит реализацию POSIX Thread library, т.е. -lpthread указывать не надо
- libpthread.so сохранена для совместимости со старыми сборочными скриптами

Еще о сборке (Linux 2.6)

- В Linux libstdc++.so содержит «слабые» (weak) определения символов POSIX Thread library.
- Поэтому программа на C++ без ключа -lphtread соберется, но работать не будет