Операционные системы и среды
Л.р.6. Управление потоками, средства синхронизации.

Цель:

Изучение подсистемы потоков (pthread), основных особенностей функционирования и управления, средств взаимодействия потоков. Практическое проектирование, реализация и отладка программ с параллельными взаимодействующими (конкурирующими) потоками.

Теоретическая и методическая часть

Вычислительные потоки в Unix (Linux), соотношение процессов и потоков.
Потоки pthread
Порождение потоков: pthread_create(). Функция потока. Виды потоков (joinable и detach) и их атрибуты.
Завершение потоков: pthread_exit() и pthread_cancel(). «Точки принудительного завершения» (cancelation points), поведение при принудительном завершения.

Мьютексы pthread_mutex) – типы мьютексов, особенности поведения, API мьютексов.
Барьеры pthread_barrier – особенности поведения, API барьеров.
Спин-блокировки pthread_spinlock – особенности поведения и применения, API спин-блокировок.
«Быстрые мьютексы» futex – виды, особенности, API «быстрых мьютексов»

Практическая часть

Общая постановка задачи:

Написать программу (программы) в соответствии с вариантом задания. Спланировать и обеспечить тестирование (демонстрацию) выполнения – для нескольких взаимодействующих потоков это может быть существенно более сложно и трудоемко.

Желательно продолжать использовать *make* (и сценарии *makefile*) для управления обработкой проекта.

Многие из вариантов имеют сходство с заданиями по дисциплине СП.

Варианты заданий:

– Многопоточная сортировка (или иные виды обработки данных)
– Многопоточная работа с файлом
– «Надежный» менеджер потоков (контроль работоспособности, восстановление выбывших потоков)
– Нагрузочная способность («стресс-тест»)
– Сравнение эффективности средств синхронизации

– Модели взаимодействия параллельных потоков
–

1 Многопоточная сортировка

Многопоточная программа, реализующая обработку достаточно большого массива данных, например его сортировку (алгоритм обработки должен допускать эффективное распараллеливание).

Типовые стадии обработки (на примере сортировки):
– разбиение массива на несколько частей (фрагментов)
– сортировка каждого фрагмента отдельным потоком
– окончательная «сборка».

Количество потоков (в т.ч. единственный) и размер массива задаются пользователем. Количество потоков выбирается не слишком большое, чтобы оставалось удобным для отображения и не провоцировало перегрузку системы.

Результат – сведения о времени выполнения для конкретной конфигурации, минимальный протокол выполнения.

2 Многопоточная работа с файлом

В целом аналогично предыдущему варианту, но для операций ввода-вывода (дисковый файл, открытый для разделяемого доступа).

Сравнение с обычной (последовательной) реализацией при тех же количествах операций и объемах передаваемых данных.

3 «Надежный» менеджер потоков

Многопоточная программа, состоящая из потоков двух типов: «менеджера» и «рабочих потоков».

Поток-«менеджер» (скорее всего, главный поток):
– создание «рабочих» потоков;
– распределение заданий и сбор результатов (в данном случае задания могут моделироваться, т.е. быть имитацией);
– контроль состояния «рабочих» потоков и при необходимости перезапуск для сохранения общего количества;
– согласованное завершение всех потоков программы.

«Рабочие» потоки (создаваемые главным):
– выполнение заданий (моделирование выполнения)
– моделирование аварий (аварийное завершение с некоторой частотой (вероятностью).

Для контроля состояния «рабочих» потоков могут служить, например, периодически устанавливаемые/сбрасываемые и проверяемые мьютексы или семафоры, для синхронизации при выполнении заданий – мьютексы, семафоры или барьеры.

Результат выполнения – протокол сеанса моделирования.

4 Нагрузочная способность («стресс-тест»)

Многопоточная программа, создающая нагрузку на систему с целью оценки влияния количества одновременно выполняющихся потоков на общую загруженность, отзывчивость, снижение производительности других программ и т.п. Алгоритм функций потоков выбирается таким, чтобы минимизировать зависимости между потоками и избежать взаимного влияния и блокировок.

Необходимо обеспечить достаточно плавное нарастание нагрузки, позволяющее наблюдать эффект и избежать аварии системы.

Имеет смысл сравнить результаты с аналогичным тестов по Windows на той же (или близкой по параметрам) технике.

Помимо аппаратных ресурсов, влияние могут оказывать административные ограничения (квоты).

5 Сравнение эффективности средств синхронизации

Программа, потоки (или большинство потоков) которой интенсивно используют средства синхронизации, с оценкой затрат времени при использовании различных ISO: семафоров (System V и/или POSIX), мьютексов (pthread_mutex) разного типа и т.д. Нужна возможность провести измерения при разных конфигурациях: количество потоков, частота обращения к ISO и т.д.

Во время измерений необходимо позаботиться о минимизации влияния побочных факторов, например операций ввода-вывода, и о создании достаточной (но не избыточной!) вычислительной нагрузки для имитации реалистичных условий выполнения.

В силу сложности задачи и множественности факторов, влияющих на результат, результат будет отражать лишь отдельные частные случаи. Программу можно будет рассматривать как заготовку инструмента для более масштабных экспериментов или для анализа других частных случаев.

6 Модели взаимодействия параллельных потоков

Реализация многопоточной программой одной из моделей взаимодействия, например «задачи об обедающих философах». Конечная цель – демонстрация (и проверка) функционирования модели и различных подходов и механизмов для обеспечения корректности этого функционирования.

Изменяемые параметры модели: количество взаимодействующих участников (блоков), интенсивность событий, характеристики случайных величин (интервалы, задержки и т.п.), выбор логики разрешения конфликтов, величины тайм-аутов и т.д.

Результат выполнения – протокол сеанса моделирования, извещения об обнаруженных особых состояниях, например тупиках.