Шилов Николай Вячеславович
Кандидат физико-математических наук, старший научный сотрудник Института систем информатики имени А.П. Ершова Сибирского отделения РАН.

Шилова Светлана Олеговна
Инженер Института систем информатики им. А.П. Ершова Сибирского отделения РАН.

Что такое «параллельное программирование»

Всякое направление в науке, технике и технологии характеризуется прежде всего подходом к постановке и методам решения своих задач. Не является исключением и параллельное программирование – современное активно развивающееся направление научных исследований, вычислительной техники и технологий. Статья знакомит с основными понятиями параллельного программирования, его методами постановки и решения задач на примере головоломки.

О параллельном жаренье бифштексов

В параллельном программировании постановка задачи и её решение предполагают, что несколько вычислительных «потоков» или/и «процессов» могут работать одновременно («параллельно») и ускорение расчётов получают за счёт использования имеющихся вычислительных ресурсов. Здесь «ускорение» – это «во сколько раз быстрее» по сравнению с последовательным («непараллельным») или «менее параллельным» решением, а не «скорость изменения скорости», как в физике. Поясним это определение на примере решения следующей известной головоломки о бифштексах.

На сковородку помещается 2 бифштекса. Бифштекс надо поджарить с двух сторон, а поджаривание каждого бифштекса с одной стороны занимает 5 минут. За сколько минут можно приготовить 3 бифштекса?

Самое первое решение – готовить их по очереди; тогда на приготовление каждого уйдет 10 минут (т. к. обжаривать нужно с двух сторон), а всего потребуется 30 минут. Второе решение состоит в том, что сначала готовят синхронно (т. е. полностью одновременно) первую пару бифштексов, а потом отдельно – третий бифштекс; такой вариант займёт 20 минут. Но есть и третье ре-

шение. Дадим бифштексам имена «A», «B» и «C», а их сторонам присвоим индексы «1» и «2». Мы считаем, что эта операция не заняла времени, но теперь мы можем говорить, например, о стороне «B2», т. е. о второй стороне второго бифштекса. Далее алгоритм таков:

1) поджарим одновременно стороны «A1» и «B1»;

2) поджарим одновременно стороны «B2» и «C1»;

3) поджарим одновременно стороны «C2» и «A2».

Этот алгоритм организует «конвейер» приготовления бифштексов: стороны бифштексов A1, B1, B2, C1, C2, A2 как бы лежат на ленте конвейера, которая движется через сковородку ( ):

(A1, B1), B2, C1, C2, A2 – первые 5 минут,

A1, B1, (B2, C1), C2, A2 – вторые 5 минут,

A1, B1, B2, C1, (C2, A2) – третьи 5 минут.

Он позволяет приготовить три бифштекса за 15 минут; следовательно, ускорение, достигнутое по сравнению с первым (последовательным) решением, равно $30/15 = 2$, а по сравнению со вторым «менее параллельным» решением (использующим синхронное приготовление пары бифштексов) равно $20/15 \approx 1{,}33$.

Потоки на двухъядерной сковородке

Поток, или тред (от англ. thread – нить) – это последовательность команд, которая «исполняется» вычислительным устройством «одновременно» с несколькими другими потоками над общей памятью. Слова «исполняются» и «одновременно» взяты в кавычки, так как у вычислительного устройства может не хватать средств для действительно одновременного исполнения всех потоков. Потоки просто приняты к исполнению, а вычислительное устройство по очереди по частям выполняет все действия, предусмотренные в этих потоках. (В таком случае говорят о «псевдопараллелизме», т. к. нет реального одновременного исполнения разных действий, или об «интерливинговом параллелизме», используя кальку английского термина «interleave», означающего прокладывание между листами книги (гербария, например).

Частный случай многопоточного программирования – это обычное императивное программирование, когда единственное вычислительное устройство занято одним потоком.

Обычно многопотоковый алгоритм реализуется на «многоядерном процессоре» – вычислительном устройстве, состоящем из нескольких «ядер», т. е. обычных процессоров, работающих над общей памятью. При этом надо иметь в виду, что одно ядро нужно для планировщика – алгоритма, который отвечает за загрузку других ядер (т. е. командует, какому ядру какой поток и когда выполнять).

В терминах многопоточного параллельного программирования головоломку о бифштексах можно интерпретировать так: сковородка – это двухъядерный процессор, бифштексы – три потока «A», «B» и «C», каждый поток состоит из двух последовательных команд «поджарить со стороны 1» и «поджарить со стороны 2». Планировщик здесь – человек, который составляет алгоритм приготовления бифштексов. Умный планировщик – и бифштексы приготовлены на «двухъядерной» сковородке за 15 минут; если же с планировщиком не повезло, то приготовление может занять и 20, и даже 30 минут.

Потоки могут разделяться и сливаться, обычно для этого используют конструкцию fork-join (от англ. fork – развилка и join – соединение) и разделитель «‖». Например, описанный трёхпоточный алгоритм приготовления трёх бифштексов можно записать так:

```
begin fork
поджарить A1; поджарить A2 ‖
поджарить B1; поджарить B2 ‖
поджарить C1; поджарить C2
join end
```

Чтобы процесс пошёл...

Теперь познакомимся с процессами. Процесс – это программный шаблон (или матрица), с которого можно сделать сколько угодно программ-копий; они называются экземплярами процесса. Можно образно представить, что процесс – это чертёж робота, а экземпляры процесса – это роботы, изготовленные по данному чертежу.

Каждый процесс имеет имя и две числовые характеристики: число ini экземпляров, которые создаются при запуске параллельной программы, и число max, ограничивающее сверху количество экземпляров, которые могут существовать одновременно. Экземпляры процессов могут порождать новые экземпляры процессов по именам процессов.

Каждый экземпляр процесса получает свой уникальный идентификатор (адрес или пин) и свою собственную память (она недоступна другим экземплярам ни того же процесса, ни других процессов). Экземпляры процессов общаются между собой посредством обмена сообщениями: посылка сообщения происходит по идентификатору экземпляра-получателя сообщения (экземпляр-отправитель должен знать, кому он отправляет сообщение).

Каждый сеанс связи состоит в том, что экземпляр-отправитель отправляет сообщение, и экземпляр-получатель получает сообщение только одновременно, как бы во время рукопожатия или рандеву; если отправитель или получатель ещё не готовы отправить или получить сообщение (не пришли на свидание), то партнер ждёт; но сразу после успешного свидания оба участвовавших экземпляра продолжают работу по своей программе.

Обычно процессный параллелизм реализуется в «многопроцессорной среде» или «кластере» (от англ. cluster – скопление) – совокупности процессоров (или компьютеров), соединённых каналами для передачи сообщений. Каждый из процессоров имеет свою собственную (локальную) память, причём один из этих процессоров осуществляет функции диспетчера по выделению пинов для процессов и реализует алгоритм планировщика загрузки процессоров (когда, кому и какой процесс выполнять), а ещё несколько процессоров – маршрутизаторы сообщений (какое сообщение кому передать). В идеальном случае число процессоров (или элементов кластера) – по одному на каждый экземпляр процесса, который создаётся во время работы программы; но в случае, если процессоров не хватает, планировщику приходится «загружать» и «выгружать» экземпляры процессов в нужном порядке в имеющиеся процессоры.

Процесс поджарки бифштекса

Ещё раз вернёмся к головоломке про бифштексы и дадим ей процессную интерпретацию. Теперь сковородка – это уже не двухъядерный процессор, а вычислительная среда, состоящая из двух отдельных процессоров; три бифштекса – это три экземпляра одного процесса «поджарить бифштекс», который состоит из двух последовательных команд «поджарить сторону 1» и «поджарить сторону 2», а планировщик – по-прежнему алгоритм, который отвечает за загрузку и выгрузку экземпляров процессов по процессорам. Псевдокод процесса «поджарить бифштекс» можно записать так:

```
process поджарить_бифштекс: ini=3, max=2;
поджарить сторону 1; поджарить сторону 2
end.
```

В момент запуска такой программы «создаётся три бифштекса», вернее – три экземпляра процесса «поджарить бифштекс». Как и в многопотоковом случае, «умный» планировщик организует выполнение этой программы за 15 минут, «загружая» и «выгружая» 3 экземпляра процесса в нужном порядке в 2 процессора.

Заключение

Мы надеемся, что в этой статье нам удалось познакомить читателей не только с параллельным жареньем бифштексов, но и с основными понятиями параллельного программирования. Для желающих перейти к знакомству с современными технологиями параллельного программирования мы рекомендуем для начального чтения статью [1]. А для лучшего усвоения материала нашей статьи мы предлагаем всем желающим выполнить небольшое домашнее задание и ответить на следующие вопросы.

1. Можно ли приготовить три бифштекса быстрее, чем за 15 минут, если каждый бифштекс надо поджарить с двух сторон, поджаривание каждого бифштекса с одной стороны занимает 5 минут, а на сковородку помещается только 2 бифштекса?

2. Каково будет ускорение по сравнению с синхронным попарным методом, если организовать конвейер для для приготовления N бифштексов при тех же условиях (т. е. каждый бифштекс надо обжарить с двух сторон, поджаривание каждого бифштекса с одной стороны занимает 5 минут, а на сковородку помещается только 2 бифштекса)? Говорят, что что параллельный алгоритм хорошо масштабируется, если достигнутое ускорение сохраняется при увеличении размерности задачи. Так вот, является ли конвейер хорошо масштабируемым алгоритмом приготовления N бифштексов?

3. «Стихотворный» конкурс. Помните детскую загадку «А и Б сидели на трубе»? Алгоритм приготовления бифштексов так и просится в стихи: «А и Б лежат в сковороде...» Так вот, объявляется конкурс на лучшее стихотворное переложение алгоритма приготовления бифштексов. Начало может быть любое (не обязательно «А и Б лежат в сковороде...»). Объём – не более 24 строк. Приз – три поджаренных бифштекса при личной встрече с авторами.

Литература

1. *Чернышов А.* Введение в технологии параллельного программирования. Доступна на http://software.intel.com/ru-ru/articles/writing-parallel-programs-a-multi-language-tutorial-introduction/. (Проверено 4 февраля 2011 г.)