Вычисления на CPU средствами Matlab Parallel Computing Toolbox: конструкция spmd

САМОСТОЯТЕЛЬНАЯ РАБОТА

http://www.kimrt.ru
Онлайн-курс
Суперкомпьютерные технологии в задачах моделирования

Немного о том, что такое spmd

The single program multiple data или конструкция с множественными данными для одной программы (*spmd*) позволяет плавно чередовать последовательное и параллельное программирование. Оператор *spmd* позволяет определить блок кода для одновременного запуска на нескольких рабочих процессах [1].

Аспект «одной программы» в *spmd* означает, что один и тот же код выполняется на нескольких рабочих процессах. Вы запускаете одну программу в Matlab и те ее части, которые помечены как блоки *spmd*, выполняются на рабочих процессах. Когда блок *spmd* завершен, ваша программа продолжает работать далее в последовательном режиме.

Аспект «множественных данных» означает, что даже несмотря на то, что оператор *spmd* выполняет одинаковый код на всех рабочих процессах, каждый рабочий процесс может иметь разные уникальные данные для этого кода. Таким образом, несколько наборов данных могут обрабатываться несколькими рабочими процессами [1].

Типичными приложениями, подходящими для *spmd*, являются те, которые требуют одновременного выполнения программы на нескольких наборах данных, когда требуется связь или синхронизация между рабочими процессами. Некоторые распространенные случаи:

1. Программы, выполнение которых занимает много времени — *spmd* позволяет нескольким рабочим процессам одновременно производить расчёты.

2. Программы, работающие с большими наборами данных — *spmd* позволяет распределять данные между несколькими рабочими процессами.

Отличие в работе *parfor* и *spmd* заключается в том, что *parfor* просто распределяет вычислительную нагрузку на несколько рабочих процессов, не давая доступ отдельно к каждому, а *spmd* позволяет обращаться к отдельным процессам по их индексу и использовать их по-разному.

Синтаксис spmd

Общий вид выражения для работы *spmd* следующий

```
spmd
    <операторы>
end
```

Блок кода, представленный <операторами>, выполняется параллельно одновременно на всех рабочих процессах в параллельном пуле. Если нужно ограничить выполнение только частью этих рабочих процессах, необходимо указать, сколько процессов нужно запустить:

```
spmd (n)
    <statements>
end
```

Этот оператор требует, чтобы *n* рабочих процессов исполнили код *spmd*. Число *n* должно быть меньше или равно количеству рабочих процессов в открытом параллельном пуле. Если пул достаточно велик, но *n* рабочих процессов недоступны, инструкция ожидает, пока не будет доступно достаточное количество рабочих процессов. Если *n* равно 0, оператор *spmd* не использует рабочих процессов и выполняется последовательно, как если бы в данный момент не выполнялся параллельный пул [1].

Например, создадим матрицу случайных чисел 4x4 на 3-х рабочих процессах.

```
spmd (3)
    R = rand(4,4);
end
```

В отличие от цикла *parfor*, рабочие процессы, используемые для оператора *spmd*, имеют уникальное значение *labindex*. Это позволяет указать код, который будет выполняться только на определенных рабочих процессах, или настроить выполнение, обычно с целью доступа к уникальным данным [1].

Например, можно создать массивы разного размера в зависимости от *labindex*:

```
spmd (3)
    if labindex==1
        R = rand(9,9);
      else
        R = rand(4,4);
    end
end
```

Пример программы с использованием spmd

Запускаем *spmd* на 3 рабочих процессах [2].

```
spmd(3)
  q = magic(labindex + 2);
end
```

Визуализируем результаты

```
figure
```

```
subplot(1,3,1), imagesc(q{1});
subplot(1,3,2), imagesc(q{2});
subplot(1,3,3), imagesc(q{3});
```

При необходимости мы можем завершить работу параллельного пула

```
delete(gcp);
```

Здесь *gcp* – текущий пул параллельных вычислений.

В данном примере используется функция *magic*, которая возвращает матрицу размером *n* на *n*, составленную из целых чисел от 1 до n^2 с равными суммами строк и столбцов. Другая функция *imagesc* создаёт картинку из матрицы, а *subplot* позволяет разместить несколько окон графиков на одной панели вывода [2].

Задача этого примера в том, чтобы показать, что в зависимости от значения *labindex* могут получаться и обрабатываться уникальные данные на различных рабочих процессах.

Результаты выполнения программы видны на рис.1.

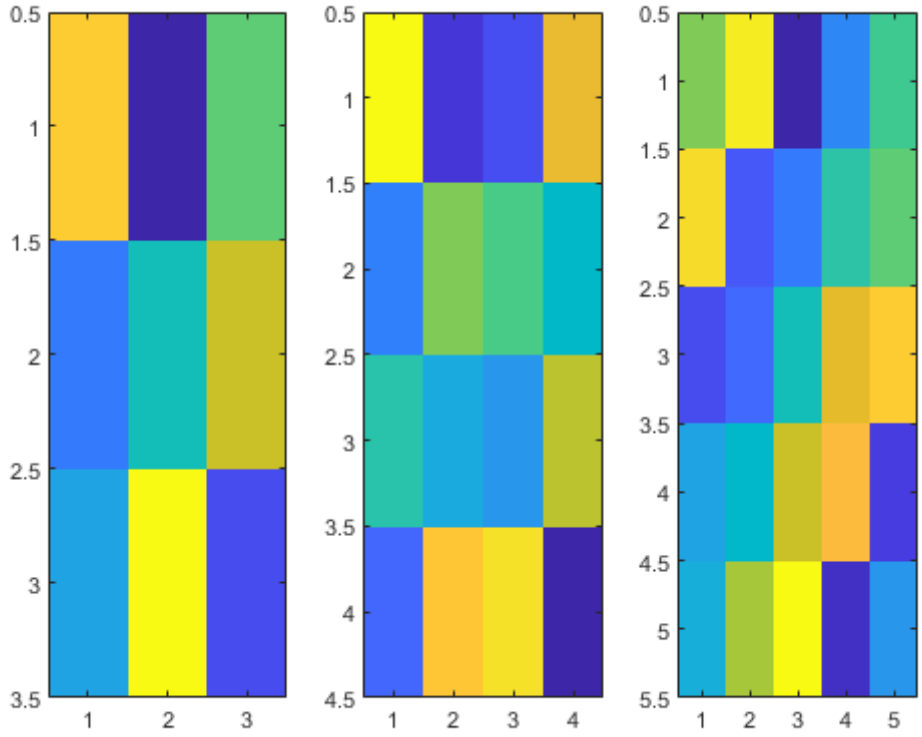

Рис.1. Демонстрация результатов работы программы, использующей *spmd*.

http://www.kimrt.ru/index/course_stm/0-24

Проект реализуется победителем Конкурса на предоставление грантов преподавателям магистратуры 2020/2021 благотворительной программы «Стипендиальная программа Владимира Потанина» Благотворительного фонда Владимира Потанина.

Источники

1. Run Single Programs on Multiple Data Sets - MATLAB [Электронный ресурс] // MathWorks - URL: https://www.mathworks.com/help/parallel-computing/execute-simultaneously-on-multiple-data-sets.html , – (дата обращения: 10.05.2022).
2. Execute code in parallel on workers of parallel pool - MATLAB [Электронный ресурс] // MathWorks - URL: https://www.mathworks.com/help/parallel-computing/spmd.html , – (дата обращения: 12.05.2022).