ОДИН ПОДХОД К ОНЛАЙН-ВИЗУАЛИЗАЦИИ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ

П. А. Васёв
Институт математики и механики им. Н. Н. Красовского
Уральского отделения Российской академии наук, г. Екатеринбург

Параллельные вычисления играют ключевую роль в компьютерном моделировании явлений. Онлайн-визуализация позволяет визуализировать состояние вычислений и управлять ими. Вопрос заключается в том, как реализовать онлайн-визуализацию. В докладе предлагается оригинальный подход. Он опирается на следующие положения:

1. Визуализация реализуется через вычисление. Действительно, операции фильтрации и преобразования данных, построение геометрических образов, растеризация — вполне математические операции.

2. Ввиду большого объёма визуализируемых данных их визуализацию необходимо реализовывать параллельно. К задачам визуализации можно применять те же методы распараллеливания, что и в общем случае при моделировании явлений.

3. Визуализируемые данные располагаются в оперативной памяти вычислительных узлов. Необходима такая технология вычислений, чтобы алгоритмы визуализации имели прямой доступ к этим данным (на чтение).

Исходя из этих положений, подход рассматривает параллельное программирование в целом, а задачу их онлайн-визуализации как частный случай. В основе подхода лежат следующие понятия и модель взаимодействия.

1. **Исполнитель** – процесс в терминах операционной системы, который служит местом исполнения вычислительных процессов. Исполнителю соответствует языковая среда и оборудование. Например один вид исполнителя может выполнять процессы для обычного процессора, заданные в двоичных кодах; другой – в среде Python, третий – для GPU, и т. п. Исполнители запускаются на узлах вычислительного кластера.

2. **Вычислительный процесс** – это логический процесс, который проводит какие-либо вычисления, преобразования данных и т. п. Он запускается на заданном исполнителе. Исполнитель выполняет множество вычислительных процессов. Это сделано для того, чтобы вычислительные процессы могли иметь прямой доступ к оперативной памяти друг друга.

3. **Ячейка** — это коммуникационная сущность, которую можно открыть на чтение или на запись. Вводится операция записи сообщения и чтения сообщения. Сообщение это некая произвольная структура данных.

4. **Связь между ячейками** – это особый процесс, который обеспечивает передачу информации между двумя ячейками. Один вычислительный процесс может открыть некую ячейку на запись и записывать в неё информацию, другой процесс открыть другую ячейку на чтение и читать информацию. Если эти ячейки соединить связью – будет происходить передача информации от первой ячейки ко второй и таким образом – от первого процесса ко второму. Вводится окружение, которое сопоставляет ячейки идентификаторам. Открытие ячеек и наладку связей между ними можно осуществлять по их идентификаторам.

Представленные понятия позволяют определять параллельные вычисления для разных задач. Можно писать программу, которая будет запускать на узлах кластера исполнители, размещать в них вычислительные процессы, налаживать связи между ячейками и т. п. Однако выяснилось, что эту модель можно дополнить надстройкой, которая позволяет мыслить о параллельных вычислениях удобнее и на более высоком уровне абстракции. Эта надстройка вдохновлена и использует идеи, изложенные в работе [1]. Вводятся понятия:

1. **Макропроцесс** – вид вычислительного процесса, который отличается тем, что запускает множество подпроцессов на других исполнителях. Это позволяет мыслить о процессе, а в то же время управлять целым вычислительным полем. Макропроцессы являются аналогией объектов Paraview Server Manager [2]. Отличие заключается в том, что объекты Server Manager в Paraview дублируются на все исполнители, а макропроцессы – сами решают, на каких исполнителях какие подпроцессы следует запускать.

2. **Порт** – это коммуникационная сущность, которая является множеством ячеек с дополнительной структурой, соответствующей распределению данных. Например, запускается некоторый макропроцесс с подпроцессами на *K* исполнителях. Пусть каждый такой подпроцесс имеет свою ячейку входных данных. Объединим идентификаторы этих ячеек в массив и получим входной порт для макропроцесса. Аналогично и с выходным портом. У макропроцесса может быть несколько входных и выходных портов.

3. **Связь между портами** – это особый процесс, который обеспечивает передачу информации между двумя портами. Связываемые порты должны обладать одинаковой структурой и количеством ячеек. Связь между портами реализуется как набор связей между соответствующими ячейками портов, один к одному, аналогично как в электрике соединяют многожильные кабели. Автоматической редукции *MxN*, как в проекте ADIOS2, не предполагается: для этого можно создать макропроцесс по преобразованию.

Параллельная программа определяется обычной программой, задача которой – формировать граф из макропроцессов и связей между их портами. Например, программу можно оснастить графическим интерфейсом, и реагировать на управляющие воздействия пользователя соответствующей коррекцией графа. Программа эта по сути есть управляющий процесс, который управляя графом определяет вычисления, происходящие на кластере.

Литература.

1. Надуев, А. Г., Черевань, А. Д., Лебедева, А.С., Архитектура программного модуля "ЛОГОС ПЛАТФОРМА" // Вопросы атомной науки и техники. Серия: Математическое моделирование физических процессов. 2022. – № 4. – С. 55-63. – DOI 10.53403/24140171_2022_4_55.
2. Andy Cedilnik, Berk Geveci, Kenneth Moreland, James Ahrens, and Jean Favre, Remote Large Data Visualization in the ParaView Framework. In *Eurographics Parallel Graphics and Visualization 2006*, May 2006, pp. 163–170. DOI 10.2312/EGPGV/EGPGV06/163-170.