Параллельная реализация метода конечных элементов с использованием GPU

Шиловский Максим Викторович
Новосибирский государственный технический университет (Новосибирск), Россия
e-mail: max-in-3d@yandex.ru

Одним из этапов конечноэлементных методов является сборка глобальной матрицы СЛАУ из локальных матриц. Обычно локальные матрицы рассчитываются аналитически, однако достаточно часто возникает необходимость численного интегрирования при генерации локальных матриц. В частности, численное интегрирование используется в случае, если базисные функции трудно или невозможно интегрировать аналитически. Использование численного интегрирования, как правило, ведет к увеличению времени генерации глобальной матрицы по сравнению с аналитическим рассчетом, что в случае нестационарной или нелинейной задачи приводит к существенному увеличению времени расчета.

Особенностью конечноэлементных методов является независимость локальных матриц, что, теоретически, позволяет ускорить процесс сборки за счет их параллельной генерации. В данной работе рассматривается генерация локальных матриц для классического метода конечных элементов и разрывного метода Галеркина на GPU с использованием технологии OpenCL. Методы конечных элементов применялись к решению задачи о фильтрации жидкости в пористой среде. Для ускорения работы на GPU генерировались и сохранялись локальные матрицы и матрицы перехода, а окончательная сборка осуществлялась на CPU, что позволило избежать блокировок при реализации кернелов. Численные эксперименты проводились на компьютере с установленными видеокартами NVidia GeForce GT 650M и Intel HD4000 и центральным процессором Intel Core i5. Сравнение скорости работы проводилось с реализацией метода конечных элементов с использованием OpenMP. Расчеты проводились с использованием как одинарной, так и двойной точности.

При использовании двойной точности, расчет на видеокарте NVidia выполнялся в несколько раз дольше, чем аналогичная программа на CPU, что согласуется с заявленной низкой производительностью данной видеокарты при расчетах с двойной точностью. Стоит отметить, однако, что при запуске программы с использованием OpenCL и CPU в качестве устройства выполнения кернела, время работы, как правило, было меньше, чем у программы с использованием OpenMP. При использовании float, время генерации СЛАУ с использованием GPU было до двух раз меньше, чем у референсной программы. В то же время, точность вычислений заметно падала и в некоторых случаях приводила к увеличению погрешности решения на порядок по сравнению с референсной программой.