Руководство пользователя HISS

Цыбулин Иван

13 февраля 2013 г.

1 О HISS

HISS (Hybrid Iterative Sparse Solver) — библиотека для решения распределенных систем линеных уравнений разреженной структуры на графических ускорителях семейства CUDA. Библиотека разрабатывалась как простая замена библиотеке Aztec, позволяющая использовать вычислительные мощности гибридных кластеров с графическими ускорителями CUDA.

2 Терминология

Предполагается, что разреженная система получается в результате некоторой дискретизации пространственной области. При этом область описывается большим числом неизвестных переменных, сгруппированных некоторым образом в домены. Каждый домен имеет свой номер — ранг. С каждым доменом связан ровно один вычислительный процесс (MPI), имеющий в своем распоряжении один графический ускоритель. Область целиком разбита на непересекающиеся домены. С каждой неизвестной связано ровно одно линейное уравнение, включающее (возможно) кроме нее еще некоторое количество неизвестных, расположенных в пределах некоторого шаблона. При этом эти неизвестные могут оказаться в другом домене.

Если некоторые неизвестные в домене имеют шаблоны, целиком в нем не лежащие, то вместо неизвесных из других доменов вводятся теневые (ghost) переменные. Каждая теневая переменная имеет локальный номер в домене, которому принадлежит, а также знает ранг домена и локальный номер той неизвестной, которую заменяет (будем называть ее прообразом).

По отношению к каждому домену неизвестная может быть:

- собственной (self) - принадлежать домену
- теневой (ghost) - не принадлежать домену, но принадлежать шаблону какой-то неизвестной в домене
- граничной (border) - собственной и теневой для другого домена
- внутренней (inner) - собственной, но не граничной
- чужой (foreign) - не принадлежать ни домену, ни шаблону какой-либо из неизвестных в домене

Для заданного домена, все теневые неизвестные могут быть сгруппированы по доменам, к которым принадлежат их прообразы. Теневые неизвестные также могут иметь прообразы из того же домена. (это может быть удобно для "закольцованных" доменов).

3 Способ хранения данных

3.1 Хранение векторов

Все вектора хранятся распределенно. На каждом процессе расположена часть вектора, состоящая из собственных (n_s штук) и теневых (n_g штук) переменных. Теневые переменные сгруппированы по рангам процессов, на которых находятся их прообразы.

Значения переменных упакованы в массиве следующим образом:

- Собственные переменные домена с рангом $rank$
- Теневые переменные от домена с рангом 0
- Теневые переменные от домена с рангом 1
- $\vdots$
- Теневые переменные от домена с рангом $rank - 1$

- Теневые переменные от домена с рангом $rank + 1$
- $\vdots$
- Теневые переменные от домена с рангом $size - 1$

$$x^{(s)} = \begin{array}{|c|c|c|c|c|c|c|c|}\hline self^{(s)} & ghost_0^{(s)} & ghost_1^{(s)} & \dots & ghost_{s-1}^{(s)} & ghost_{s+1}^{(s)} & \dots & ghost_{p-1}^{(s)} \\ \hline \end{array}$$

3.2 Храниение информации о портрете матрицы

Индекс s означает ранг данного домена. Для каждого домена хранится отображение локальные номера теневых переменных $\mapsto$ локальные номера прообразова в своих доменах

$$\forall i \in ghost_q^{(s)}, \quad map_q^{(s)}[i] = j \Rightarrow j \in self^{(q)},$$
$$x^{(s)}.ghost_q[i] = shadow(x^{(q)}.self[j])$$

Практически, портрет матрицы строится двумя вызовами:

- addLocal(i,j) - добавление в уравнение для собственной переменной i собственной переменной j. i и j в локальной нумерации собственных переменных
- addRemote(i,j,q,rj) - добавление в уравнение для собственной переменной i теневого элемента j. i в локальной нумерации собственных переменных. j в локальной нумерации теневых элементов $ghost_q^{(s)}$. rj в локалькой нумерации собственных переменных домена q. Добавляется отображение $map_q^{(s)}(j) := rj$

3.3 Хранение матрицы

Матрица системы хранится на каждом процессе в виде двух разреженных блоков (S и S). Блок S имеет размер $n_s \times (n_s + n_b + n_e)$ Блок B имеет размер $n_b \times (n_s + n_b + n_e)$