Метод решения задачи сильной отделимости для многопроцессорных систем с массовым параллелизмом*

А.В. Ершова

Задача разделения двух выпуклых непересекающихся многогранников имеет большое значение теоретического и прикладного характера в распознавании образов, включающем задачи дискриминации, классификации и др. [1].

Задача сильной отделимости – это задача нахождения слоя наибольшей толщины, разделяющего выпуклые многогранники M и N $\left(M = \{x | Ax \le b\} \ne \emptyset, \quad N = \{x | Bx \le d\} \ne \emptyset, M \cap N = \emptyset\right)$. Сильная отделимость, по существу, эквивалентна задаче отыскания расстояния между M и N в смысле метрики $\rho(M, N) = \min\{\|x - y\| \mid x \in M, y \in N\}$.

В работе [2] описан последовательный итерационный алгоритм решения задачи сильной отделимости на базе фейеровских отображений.

Рассмотрим метод построения параллельного алгоритма решения данной задачи, в основе которого лежит подход разбиения на подвекторы. Пусть $\varphi(x)$ - произвольное фейеровское отображение относительно множества M, $\varphi \in \{R^n \to R^n\}$. Разобьем вектор $x \in R^n$ на r подвекторов $x = [x_1, \ldots, x_r]$, где $x_i \in R^{n_i}$ $(i = 1, \ldots, r)$ и $R^n = R^{n_1} \times \ldots \times R^{n_r}$. Обозначив через $\pi_i(x)$ проекцию x на R^i, определим отображения $\varphi_i(x) = \pi_i(\varphi(x)) + \sum_{j \ne i} \pi_j(x)$ и положим $\overline{\varphi}(x) = \alpha \sum_{i=1}^{r} \pi_i(\varphi_i^t(x))$, $0 < \alpha < 1$ при некотором фиксированном натуральном t. При определенных условиях отображение $\overline{\varphi}$ будет M -фейеровским. Таким образом, на базе одного фейеровского отображения мы сконструировали другое, обладающее большим ресурсом параллелизма. Действительно, значения $\varphi_i^t(x)$ могут вычисляться независимо друг от друга для различных $i = 1, \ldots, r$. При этом мы получаем две степени свободы для регулировки баланса загрузки процессорных узлов. Во-первых, увеличивая t, мы можем повышать вычислительную нагрузку на процессорный узел между двумя соседними итерациями, вычисляющими $\overline{\varphi}(x)$. Во-вторых, мы можем произвольным образом перераспределять координаты вектора x между процессорами.

Данный метод был реализован в виде программы на языке C++ с использованием пакета MPI-2. Проведенные вычислительные эксперименты показали, что алгоритм может допускать эффективное масштабирование на тысячи процессорных узлов.

Вычислительные эксперименты проводились на сконструированных модельных масштабируемых задачах *Model-n*. Для всех таких задач можно легко аналитически вычислить точное значение толщины максимального разделяющего слоя. Поэтому они хорошо подходят для проверки корректности алгоритма, подбора оптимальных параметров и исследования масштабируемости. В серии вычислительных экспериментов была исследована зависимость количества итераций от количества независимых без обменов итераций k, которое варьировалось от 1 до 100, обе задачи решались для размерностей n = 10, 50, 100. Графики показывают (рис. 1 и рис. 2), что для всех размерностей количество итераций уменьшается с увеличением количества независимых без обменов итераций, принимая горизонтально-асимптотический характер для значений k, больших 50. Проведенные эксперименты демонстрируют, что в большинстве случаев хорошим выбором является значение k = 50.

* Работа поддержана грантом РФФИ № 09-01-00546а и грантом Роснауки по поддержке ведущих научных школ НШ-5595.2006.1

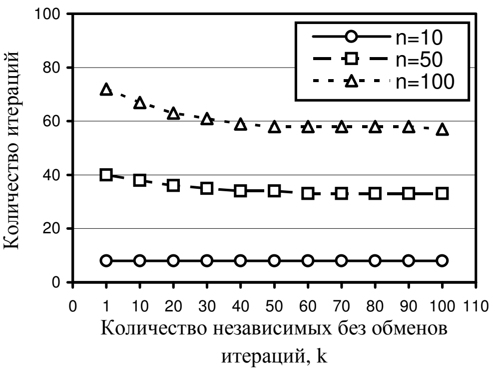

Рис. 1. Зависимость количества итераций от количества независимых без обменов итераций *k* для задачи №1 из серии *Model-n*.

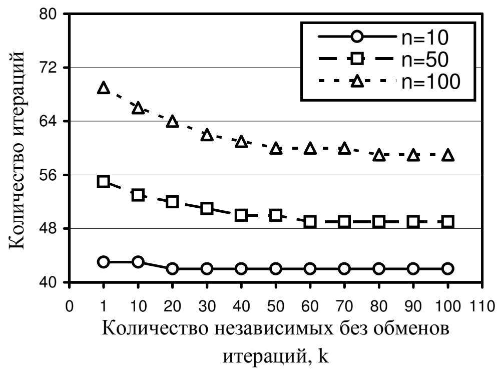

Рис. 2. Зависимость количества итераций от количества независимых без обменов итераций *k* для задачи №2 из серии *Model-n*.

Литература

1. Еремин И.И. Фейеровские методы сильной отделимости выпуклых полиэдральных множеств // Известия вузов. Сер. математика. -2006. -№ 12. -С. 33-43.
2. Ершова А.В. Алгоритм разделения двух выпуклых непересекающихся многогранников с использованием фейеровских отображений // Системы управления и информационные технологии. -2009. -№1(35). -С. 53-56.