

Применение S-технологии для решения задач линейного программирования на многопроцессорных системах с массовым параллелизмом*

Ю.С. Асфандиярова, И.М. Соколинская

На основе метода, описанного в [1], был составлен следующий алгоритм для решения задач линейного программирования. В основе алгоритма лежит S-технология. Решение исходной задачи линейного программирования находится в результате итерационного процесса с циклическим применением следующих отображений для последовательности $\{[x_k, u_k]\}_{k=1}^{+\infty}$:

$$\varphi_1(x) = x - (\lambda_1/\delta_1) \sum_{j=1}^m l_j^+(x) a_j, \quad l_j^+(x) = \max\{0, (a_j, x) - b_j\}, \quad \delta_1 = \sum_{j=1}^m \|a_j\|^2, \quad \lambda_1 \in (0, 2).$$

$$\varphi_2(u) = u + (\lambda_2/\delta_2) \sum_{i=1}^n h_i^+(u) h_i, \quad h_i^+(u) = \max\{0, (c_i, u) - h_i\}, \quad \delta_2 = \sum_{i=1}^n \|h_i\|^2, \quad \lambda_2 \in (0, 2).$$

$$\varphi_3(x, u) = [x, u]^T - \frac{(c, x) - (b, u)}{\|c\|^2 + \|b\|^2} [c, -b]^T.$$

Особенность алгоритма заключается в том, что исходные векторы x и u разбиваются на подвекторы: вектор x на r подвекторов $x = [x_1, x_2, \dots, x_r]$, где $x_i \in R^{n_i}$ ($i = 1, \dots, r$) и $R^n = R^{n_1} \times \dots \times R^{n_r}$, а вектор u на s подвекторов $u = [u_1, u_2, \dots, u_s]$, где $u_j \in R^{m_j}$ ($j = 1, \dots, s$) и $R^m = R^{m_1} \times \dots \times R^{m_s}$. Обозначив через $\pi_{1i}(x)$ проекцию x на R^{n_i} и через $\pi_{2j}(u)$ проекцию u на R^{m_j} , определим отображения $\varphi_{1i}(x) = \pi_{1i}(\varphi_1(x)) + \sum_{j \neq i} \pi_{1j}(x)$ и

$$\varphi_{2j}(u) = \pi_{2j}(\varphi_2(u)) + \sum_{i \neq j} \pi_{2i}(u), \quad \text{положим } \bar{\varphi}_1 = \alpha \sum_{i=1}^r \pi_{1i}(\varphi_{1i}^t(x)),$$

$$\bar{\varphi}_2 = \alpha \sum_{i=1}^s \pi_{2i}(\varphi_{2i}^t(u)) \quad 0 < \alpha < 1 \quad \text{при некотором фиксированном натуральном } t \quad (\text{возможно}$$

разным для $\bar{\varphi}_1$ и для $\bar{\varphi}_2$). Значения $\varphi_{1i}^t(x)$ и $\varphi_{2j}^t(u)$ могут вычисляться параллельно, независимо друг от друга для $i = 1, \dots, r$ и $j = 1, \dots, s$, т.е. отображения φ_{ij}^t образуют $r + s$ параллельных процессов. При этом мы получаем две степени свободы для регулирования загруженности процессоров: во-первых, увеличивая t , можно повышать вычислительную нагрузку на процессорный узел между соседними итерациями, во-вторых, изменяя количество подвекторов ($r + s$) и количество координат в каждом подвекторе мы можем произвольным образом перераспределять координаты векторов между процессорами (см. также [2]).

На основе указанного алгоритма была разработана параллельная программа на языке C++, реализующая данный метод, для многопроцессорной системы с массовым параллелизмом. В настоящее время проводятся вычислительные эксперименты по исследованию эффективности параллельного алгоритма на задачах большой размерности, по исследованию зависимости количества итераций, времени выполнения программы и числа обменов между процессорами от параметров алгоритма.

Литература

1. Еремин И.И. Теория двойственности в линейной оптимизации. Челябинск: «Библиотека А. Миллера», 2005.- 196 с.
2. Л.Б. Соколинский Иерархический параллелизм – новая парадигма программирования. ЮУрГУ, Челябинск.

* Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (проект 06-01-00380).