УДК 519.6 doi 10.26089/NumMet.v16r336

ПАРАЛЛЕЛЬНАЯ РЕАЛИЗАЦИЯ МАТРИЧНОГО КРЕСТОВОГО МЕТОДА

Д. А. Желтков[1], Е. Е. Тыртышников[2]

Матричный крестовый метод является быстрым методом аппроксимации матриц матрицами малого ранга, его сложность составляет $O\big((m+n)r^2\big)$ операций. Важной особенностью является то, что если матрица задана не как хранящийся в памяти массив, а как функция от двух целочисленных аргументов, то можно найти ее малоранговое приближение, вычислив лишь $O\big((m+n)r\big)$ значений этой функции. Однако в случае сверхбольших размеров матрицы или крайней затратности вычисления ее элементов аппроксимация может занимать существенное время. Ускорить метод для подобных случаев можно с помощью параллельных алгоритмов. В настоящей статье предложен эффективный параллельный алгоритм для случая одинаковой сложности вычисления любого элемента матрицы.

Ключевые слова: малоранговые аппроксимации, матричный крестовый метод, параллельные алгоритмы.

1. Введение. Матричный крестовый метод [1–5] является быстрым методом аппроксимации матриц матрицами малого ранга. Сложность метода составляет $O\big((m+n)r^2\big)$ операций, где m и n — размеры матрицы, а r — ранг приближения. Отметим, что приближение, находимое данным методом, является квазиоптимальным, т.е. близким к оптимальному r-ранговому приближению.

Одной из важнейших особенностей матричного крестового метода является то, что он не использует все элементы матрицы для нахождения приближения. Таким образом, если матрица задана не как хранящийся в памяти массив, а как функция от двух целочисленных аргументов, то можно найти ее малоранговое приближение, вычислив лишь $O\big((m+n)r\big)$ ее элементов. Это весьма полезно в случаях затратного вычисления элементов или больших размеров матриц.

Метод отлично себя зарекомендовал в ряде практических задач, в частности при решении интегральных уравнений [3, 6, 7], восстановлении МРТ-изображений (магнитно-резонансная томография) и многих других. Кроме того, рассматриваемый подход используется в качестве основы для методов аппроксимации многомерных массивов (тензоров) [8–10] и в практических задачах, решаемых с помощью тензорных методов [11, 12].

Однако в случае сверхбольших размеров матрицы или крайней затратности вычисления ее элементов аппроксимация может занимать существенное время. Ускорить метод для подобных случаев можно с помощью параллельных алгоритмов.

В настоящей статье изложен параллельный алгоритм матричного крестового метода для случая одинаковой сложности вычисления значения любого элемента матрицы. Параллельная сложность этого метода составляет $O\Big(\big(\lceil m/p\rceil+\lceil n/p\rceil\big)r^2+rp\Big)$ арифметических операций и $O\Big(\big(\lceil m/p\rceil+\lceil n/p\rceil\big)r\Big)$ вычислений значений элементов матрицы, где p — число процессоров. Каждый из процессоров осуществляет $O(r)$ обменов сообщениями с другими процессорами, длина каждого сообщения не больше r. Отметим, что получившееся в результате приближение хранится в удобном для дальнейшей работы формате — умножение вектора на матрицу в данном формате требует $O\Big(\big(\lceil m/p\rceil+\lceil n/p\rceil\big)r\Big)$ арифметических операций и всего одной коллективной операции взаимодействия процессоров с сообщениями длины r.

2. Матричный крестовый метод. В основе идеи крестового метода для матриц лежит следующий известный факт.

Лемма 1 (крестовое разложение). *Пусть: матрица* $A \in \mathbb{R}^{m\times n}$, $\operatorname{rank} A = r$, $\widehat{A} \in \mathbb{R}^{r\times r}$ *— подматрица матрицы* A, $\det \widehat{A} \neq 0$, $C \in \mathbb{R}^{m\times r}$ *— столбцы матрицы* A, *содержащие* $\widehat{A}$, $R \in \mathbb{R}^{r\times n}$ *— строки матрицы* A, *содержащие* $\widehat{A}$. *Тогда* $A = C\widehat{A}^{-1}R$.

Пусть теперь матрица A полного ранга, но существует матрица $E \in \mathbb{R}^{m\times n}$, такая, что E близка к нулю в смысле спектральной или фробениусовой нормы и $\operatorname{rank}(A-E) = k$. В [2–4] показано, что если

[1] Московский государственный университет им. М. В. Ломоносова, факультет вычислительной математики и кибернетики, Ленинские горы, 119992, Москва; аспирант, e-mail: dmitry.zheltkov@gmail.com

[2] Институт вычислительной математики РАН, ул. Губкина, д. 8, 119333, г. Москва; директор, e-mail: eugene.tyrtyshnikov@gmail.com

в качестве $\widehat{A}$ выбрать подматрицу матрицы A максимального объема (т.е. обладающую наибольшим по модулю определителем), то и матрица $A - C\widehat{A}^{-1}R$ тоже будет близка к нулю в смысле спектральной и фробениусовой норм.

Однако поиск подматрицы максимального объема — вычислительно сложная задача, поэтому на практике следует ограничиться достаточно большим, а не максимальным объемом. В этом случае матрица $A - C\widehat{A}^{-1}R$ тоже будет близка к нулю [2–4] в указанном выше смысле.

Пока мы полагали, что нам известно, какого ранга подматрицу необходимо искать, чтобы гарантировать необходимую точность, однако часто это не так. Известно, что выписанное приближение интерполяционно, т.е. в матрице $A - C\widehat{A}^{-1}R$ строки и столбцы с теми же номерами, что и строки и столбцы матрицы A, в которых содержится $\widehat{A}$, являются нулевыми. Поэтому легко предложить [2–4] следующую общую схему крестового метода с поиском необходимого ранга.

Алгоритм 1. Схема крестового метода (m, n, A, k).

$I \longleftarrow \{1, \ldots, m\}$
$J \longleftarrow \{1, \ldots, n\}$
$r \longleftarrow 0$
repeat
$r \longleftarrow r + 1$
$maxvol\,(A, k, I, J, I_*, J_*)$
$C_r \longleftarrow A^{J_*}\Big)$
$R_r \longleftarrow A_{I_*}\Big)$
$\widehat{A}_i \longleftarrow A_{I_* J_*}$
$A \longleftarrow A - C_r \widehat{A}_r^{-1} R_r$
$I \longleftarrow I \setminus I_*$
$J \longleftarrow J \setminus J_*$
until *stoppingcriteria*

Здесь $maxvol\,(A, k, I, J, I_*, J_*)$ — процедура, осуществляющая поиск подматрицы порядка k достаточно большого объема в подматрице матрицы A, содержащейся в строках I и столбцах J; номера строк и столбцов, содержащих найденную подматрицу, записываются в I_* и J_*; *stoppingcriteria* — некоторый критерий остановки.

В результате работы этого алгоритма матрица $\sum\limits_{i=1}^{r} C_i \widehat{A}_i^{-1} R_i$ будет приближать матрицу A.

В данной работе использовался крестовый метод с $k = 1$, т.е. процедура maxvol осуществляет поиск наибольшего по модулю элемента. В рассматриваемом случае C_i, $\widehat{A}_i$ и R_i являются соответственно вектором-столбцом, числом и вектором-строкой, поэтому обозначим

$$u_i = \frac{C_i}{\sqrt{|\widehat{A}_i|}}, \quad v_i = \frac{R_i \sqrt{|\widehat{A}_i|}}{\widehat{A}_i}, \quad U_k = \begin{bmatrix} u_1 & u_2 & \ldots & u_k \end{bmatrix}, \quad V_k = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_k \end{bmatrix}.$$

Тогда $A \approx UV$.

Используется следующий критерий остановки: $\varepsilon \|UV\|_F < |A_{ij}| \sqrt{(m-r)(n-r)}$, где ε — параметр, отвечающий за точность аппроксимации. Этот критерий хорош тем, что, с одной стороны, является достаточно надежным, так как для того чтобы была найдена аппроксимация с точностью, не ниже требуемой, достаточно, чтобы найденный нами элемент по модулю был не меньше среднего квадратичного, а на практике это, как правило, так. С другой стороны, он достаточно точный, и на практике ранг приближения не сильно отличается от минимального ранга, необходимого крестовому методу.

С учетом данного выбора алгоритм выглядит следующим образом.

Алгоритм 2. Крестовый метод (m, n, A).

$I \longleftarrow \{1, \ldots, m\}$
$J \longleftarrow \{1, \ldots, n\}$
$r \longleftarrow 0$
while true **do**

begin

$r \longleftarrow r+1$

$i \longleftarrow \arg\max\limits_{i\in I}\left|A_{iJ(1)} - [UV]_{iJ(1)}\right|$

$j \longleftarrow \arg\max\limits_{j\in J}\left|A_{ij} - [UV]_{ij}\right|$

if $\varepsilon\|UV\|_F >= |A_{ij}|\sqrt{(m-r)(n-r)}$ **then**

begin

$r \longleftarrow r-1$

break

end

$u \longleftarrow \dfrac{A^j - [UV]^j}{\sqrt{|A_{ij}|}}$

$v \longleftarrow \dfrac{\left(A_i - [UV]_i\right)\sqrt{|A_{ij}|}}{A_{ij}}$

$U \longleftarrow \begin{bmatrix}U u\end{bmatrix}$

$V \longleftarrow \begin{bmatrix}V \\ v\end{bmatrix}$

$I \longleftarrow I \setminus \{i\}$

$J \longleftarrow J \setminus \{j\}$

end

return (U, V, r)

Отметим, что непосредственное вычисление нормы $\|UV\|_F$ сложно и требует $O(mn)$ операций.

Однако известно, что $\|UV\|_F^2 = V^T U^T UV = V^T\left(U^T U\right)V$. Пользуясь этим равенством, вычисление данной нормы можно осуществлять за $O\left((m+n)r^2\right)$ операций. Тем не менее, вычисление нормы даже и таким способом слишком затратно, так как приведет алгоритм к сложности $O\left((m+n)r^3\right)$. Воспользуемся тем, что матрица является одноранговым обновлением матрицы, норма которой нам уже известна:

$$\left\|\begin{bmatrix}U u\end{bmatrix}\ \begin{bmatrix}V \\ v\end{bmatrix}\right\|_F^2 = \|UV\|_F^2 + \left(U^T u, V v^T\right) + \left(V v^T, U^T u\right) + \|u\|_2^2\,\|v\|_2^2.$$

Таким образом, пересчет нормы можно осуществлять за $O\left((m+n)r\right)$ операций. Общая сложность представленного алгоритма — $O\left((m+n)r^2\right)$ операций и $O\left((m+n)r\right)$ вычислений элементов матрицы.

3. Параллельный алгоритм. В предложенном в данной работе параллельном алгоритме поиск максимального элемента в строке (столбце) распределен между процессорами. Кроме того, матрицы U и V распределены в памяти между процессорами.

Пусть нумерация процессоров начинается с нуля и номер процессора хранится в нем в переменной rnk. Для простоты изложения примем, что m и n делятся нацело на количество процессоров p. Приведем параллельный алгоритм для этого случая.

Алгоритм 3. Параллельный крестовый метод (m, n, A, p, rnk).

$b_m \longleftarrow \dfrac{m}{p}$

$b_n \longleftarrow \dfrac{n}{p}$

if $rnk = 0$ **then** $J_* \longleftarrow \{1, \ldots, n\}$

$I \longleftarrow \left\{b_m rnk + 1, \ldots, b_m(rnk+1)\right\}$

$J \longleftarrow \left\{b_n rnk + 1, \ldots, b_n(rnk+1)\right\}$

$r \longleftarrow 0$

repeat

$r \longleftarrow r+1$

if $rnk = 0$ **then** $j \longleftarrow J_*(1)$

$Broadcast\,(0, j, j)$

$$Broadcast\left(\left[\frac{j-1}{p}\right], V^j, v\right)$$

$$i \longleftarrow \arg\max_{i \in I} \left|A_{ij} - [Uv]_i\right|$$

$$Gather\left(0, \left\{i, \left|A_{ij} - [Uv]_i\right|\right\}, \max_i\right)$$

if $rnk = 0$ **then** $\{i, column\_maximum\} \longleftarrow \max\limits_{k \in [1;p]} \left(max_i[k]\right)$

$Broadcast\,(0, i, i)$

$$Broadcast\left(\left[\frac{i-1}{p}\right], U_i, u\right)$$

$$j \longleftarrow \arg\max_{j \in J} \left|A_{ij} - [uV]_j\right|$$

$$Gather\left(0, \left\{j, \left|A_{ij} - [uV]_j\right|\right\}, max_j\right)$$

if $rnk = 0$ **then** $\{j, maximum\} \longleftarrow \max\limits_{k \in [1;p]} \left(max_j[k]\right)$

$Broadcast\,\big(0, \{j, maximum\}, \{j, maximum\}\big)$

$$Broadcast\left(\left[\frac{j-1}{p}\right], V^j, v\right)$$

$$u_{upd} \longleftarrow \frac{\left(A^j - Uv\right)\left[b_m rnk + 1 : b_m(rnk+1)\right]}{\sqrt{|maximum|}}$$

$$v_{upd} \longleftarrow \frac{(A_i - uV)\left[b_n rnk + 1 : b_n(rnk+1)\right]\sqrt{|maximum|}}{maximum}$$

$$U \longleftarrow \left[U u_{upd}\right]$$

$$V \longleftarrow \begin{bmatrix} V \\ v_{upd} \end{bmatrix}$$

$I \longleftarrow I \setminus \{i\}$

$J \longleftarrow J \setminus \{j\}$

if $(rnk = 0)$ **then** $J_* \longleftarrow J_* \setminus \{j\}$

until $tol\|UV\|_F < |maximum|\sqrt{(m-r)(n-r)}$

return (U, V, r)

Функция $Broadcast\,(i, a, b)$ рассылает переменную a с процессора i в переменную b всех процессоров; функция $Gather\,(i, a, b)$ собирает переменную a всех процессоров в переменную b (таким образом, b — массив из переменных a) процессора i.

Пересчет нормы текущего приближения легко осуществляется параллельно в условиях реализованного в алгоритме распределения по процессорам матриц U и V. Каждый процессор может вычислять векторы

$$x_{rnk} = u^T\left[b_m rnk + 1 : b_m(rnk+1)\right] U\left[b_m rnk + 1 : b_m(rnk+1), :\right],$$
$$y_{rnk} = v^T\left[b_n rnk + 1 : b_n(rnk+1)\right] V\left[:, b_n rnk + 1 : b_n(rnk+1)\right].$$

Находим суммы $x = \sum\limits_{rnk=0}^{p} x_{rnk}$, $y = \sum\limits_{rnk=0}^{p} y_{rnk}$, каждую с помощью одной операции коллективного взаимодействия процессоров. Вычисляем их скалярное произведение (x, y). Очевидно, что $(x, y) = \left(U^T u, V v^T\right)$.

Слагаемые $\left(V v^T, U^T u\right)$ и $\|u\|_2^2 \|v\|_2^2$, требуемые для пересчета нормы, вычисляются аналогично.

Для систем с общей памятью алгоритм аналогичен, но не требует пересылок данных и распределения матриц между процессорами. Для гибридных систем легко получить комбинацию алгоритмов.

Каждый процессор, кроме первого, в данном алгоритме выполняет $O\left(\left(\left\lceil\frac{m}{p}\right\rceil+\left\lceil\frac{n}{p}\right\rceil\right)r^2\right)$ операций и вычисляет $O\left(\left(\left\lceil\frac{m}{p}\right\rceil+\left\lceil\frac{n}{p}\right\rceil\right)r\right)$ элементов матрицы. Первый процессор дополнительно выполняет $O(rp)$ операций.

4. Тестирование. Метод реализован на языке C++ с использованием технологий MPI и OpenMP.

В качестве тестовой задачи выбрана следующая: на плоскости брались 2 квадрата со сторонами, параллельными осям координат, размера 1. Левый нижний угол одного расположен в начале координат, второго — в точке $(2, 2)$. В каждом из квадратов случайным образом, используя равномерное распределение, размещалось n точек. Координаты точек были сохранены в файл, поэтому входные данные одинаковы при заданном n. Далее вычислялась функция $F(i,j)=\dfrac{1}{\|x_i-y_j\|_2^2}$ (назовем ее функцией взаимодействия). Для полученной матрицы строилось приближение с точностью $\varepsilon=10^{-5}$. Тестирование производилось на суперкомпьютере BlueGene суперкомпьютерного комплекса МГУ им. М.В. Ломоносова. Приближалась матрица порядка 100 000. Результаты тестирование приведены в табл. 1.

Таблица 1
Параллельная производительность, тестовая матрица с быстрым вычислением элементов

Число процессоров	Время аппроксимации, с
1	1.47
2	0.72
4	0.38
8	0.21
16	0.13
32	0.08
64	0.05
128	0.03
256	0.02
512	0.02

Таблица 2
Параллельная производительность, тестовая матрица с медленным вычислением элементов

Число процессоров	Время аппроксимации, с
1	697.66
2	349.20
4	175.37
8	87.96
16	44.12
32	22.35
64	11.56
128	5.89
256	3.04
512	1.67

Отметим, что масштабируемость довольно хорошая, задача перестает ускоряться только при очень маленьком времени счета.

Кроме того, тестирование было выполнено для той же матрицы, но с замедленным в 1000 раз вычислением каждого элемента. Результаты приведены в табл. 2. В данном случае ускорение почти идеально.

5. Заключение. В настоящей статье предложен эффективный параллельный алгоритм матричного крестового метода для случая одинаковой сложности вычисления элементов матрицы. Алгоритм обладает достаточно низкой параллельной сложностью и довольно высокой степенью параллельности.

В дальнейшем планируется разработка алгоритма для случая различной сложности вычисления элементов матрицы с использованием параллельной модели организации вычислений “мастер–рабочие”, а также параллельные реализации методов, использующих матричный крестовый метод, в частности методы нахождения аппроксимации тензоров в различных форматах.

Настоящая статья базируется на работе, финансируемой Сколковским институтом науки и технологий (Сколтех), проект 081-R “Фундаментальные вычислительные технологии в научных и инженерных приложениях”.

СПИСОК ЛИТЕРАТУРЫ

1. *Горейнов С.А., Тыртышников Е.Е., Замарашкин Н.Л.* Псевдоскелетные аппроксимации матриц // Доклады РАН. 1995. **343**, № 2. 151–152.
2. *Goreinov S.A., Tyrtyshnikov E.E., Zamarashkin N.L.* A theory of pseudoskeleton approximations // Linear Algebra Appl. 1997. **261**, N 1–3. 1–21.

3. *Tyrtyshnikov E.* Incomplete cross approximation in the mosaic-skeleton method // Computing. 2000. **64**, N 4. 367–380.
4. *Goreinov S.A., Tyrtyshnikov E.E.* The maximum-volume concept in approximation by low-rank matrices // Contemporary Mathematics. 2001. **208**. 47–51.
5. *Goreinov S.A., Oseledets I.V., Savostyanov D.V., Tyrtyshnikov E.E., Zamarashkin N.L.* How to find a good submatrix // Matrix Methods: Theory, Algorithms, Applications. Hackensack: World Scientific, 2010. 247–256.
6. *Stavtsev S.L., Tyrtyshnikov E.E.* Application of mosaic-skeleton approximations for solving EFIE // PIERS 2009 Moscow Proceedings. Cambridge: The Electromagnetics Academy, 2009. 1752–1755.
7. *Stavtsev S.L.* Application of the method of incomplete cross approximation to a nonstationary problem of vortex rings dynamics // Russian Journal of Numerical Analysis and Mathematical Modelling. 2012. **27**, N 3. 303–320.
8. *Горейнов С.А.* О крестовой аппроксимации многоиндексного массива // Доклады РАН. 2008. **420**, № 4. 439–441.
9. *Oseledets I.V., Savostianov D.V., Tyrtyshnikov E.E.* Tucker dimensionality reduction of three-dimensional arrays in linear time // SIAM J. Matrix Anal. Appl. 2008. **30**, N 3. 939–956.
10. *Oseledets I., Tyrtyshnikov E.* TT-cross approximation for multidimensional arrays // Linear Algebra Appl. 2010. **432**, N 1. 70–88.
11. *Flad H.-J., Khoromskij B.N., Savostyanov D.V., Tyrtyshnikov E.E.* Verification of the cross 3D algorithm on quantum chemistry data // Rus. J. Numer. Anal. Math. Model. 2008. **23**, N 4. 329–344.
12. *Oseledets I.V., Savostyanov D.V., Tyrtyshnikov E.E.* Cross approximation in tensor electron density computations // Numer. Linear Algebra Appl. 2010. **17**, N 6. 935–952.

Поступила в редакцию
13.05.2015

A Parallel Implementation of the Matrix Cross Approximation Method

D. A. Zheltkov [1] **and E. E. Tyrtyshnikov** [2]

[1] *Lomonosov Moscow State University, Faculty of Computational Mathematics and Cybernetics; Leninskie Gory, Moscow, 119992, Russia; Graduate Student, e-mail: dmitry.zheltkov@gmail.com*

[2] *Institute of Numerical Mathematics, Russian Academy of Sciences; ulitsa Gubkina 8, Moscow, 119333, Russia; Dr. Sci., Professor, Corresponding Member of Russian Academy of Sciences, Director, e-mail: eugene.tyrtyshnikov@gmail.com*

Received May 13, 2015

Abstract: The matrix cross approximation method is a fast method based on low-rank matrix approximations with complexity $O((m+n)r^2)$ arithmetic operations. Its main feature consists in the following: if a matrix is not given as an array but is given as a function of two integer arguments, then this method allows one to compute the low-rank approximation of the given matrix by evaluating only $O((m+n)r)$ values of this function. However, if the matrix is extremely large or the evaluation of its elements is computationally expensive, then such an approximation becomes timeconsuming. For such cases, the performance of the method can be improved via parallelization. In this paper we propose an efficient parallel algorithm for the case of an equal computational cost for the evaluation of each matrix element.

Keywords: low-rank matrix approximations, matrix cross approximation method, parallel algorithms.

References

1. S. A. Goreinov, E. E. Tyrtyshnikov, and N. L. Zamarashkin, "Pseudoskeleton Approximations of Matrices," Dokl. Akad. Nauk **343** (2), 151–152 (1995) [Dokl. Math. **52** (1), 18–19 (1995)].

2. S. A. Goreinov, E. E. Tyrtyshnikov, and N. L. Zamarashkin, "A Theory of Pseudoskeleton Approximations," Linear Algebra Appl. **261** (1–3), 1–21 (1997).

3. E. Tyrtyshnikov, "Incomplete Cross Approximation in the Mosaic-Skeleton Method," Computing **64** (4), 367–380 (2000).

4. S. A. Goreinov and E. E. Tyrtyshnikov, "The Maximum-Volume Concept in Approximation by Low-Rank Matrices," Contemp. Math. **280**, 47–51 (2001).

5. S. A. Goreinov, I. V. Oseledets, D. V. Savostyanov, et al., "How to Find a Good Submatrix," in *Matrix Methods: Theory, Algorithms, Applications* (World Scientific, Hackensack, 2010), pp. 247–256.

6. S. L. Stavtsev and E. E. Tyrtyshnikov, "Application of Mosaic-Skeleton Approximations for Solving EFIE," in *PIERS 2009 Moscow Proc., Moscow, Russia, August 18–21, 2009* (The Electromagnetics Academy, Cambridge, 2009). pp. 1752–1755.

7. S. L. Stavtsev, "Application of the Method of Incomplete Cross Approximation to a Nonstationary Problem of Vortex Rings Dynamics," Russ. J. Numer. Anal. Math. Modelling **27** (3), 303–320 (2012).

8. S. A. Goreinov, "On Cross Approximation of Multi-Index Array," Dokl. Akad. Nauk **420** (4), 439–441 (2008) [Dokl. Math. **77** (3), 404–406 (2008)].

9. I. V. Oseledets, D. V. Savostianov, and E. E. Tyrtyshnikov, "Tucker Dimensionality Reduction of Three-Dimensional Arrays in Linear Time," SIAM J. Matrix Anal. Appl. **30** (3), 939–956 (2008).

10. I. Oseledets and E. Tyrtyshnikov, "TT-Cross Approximation for Multidimensional Arrays," Linear Algebra Appl. **432** (1), 70–88 (2010).

11. H.-J. Flad, B. N. Khoromskij, D. V. Savostyanov, and E. E. Tyrtyshnikov, "Verification of the Cross 3D Algorithm on Quantum Chemistry Data," Rus. J. Numer. Anal. Math. Modelling **23** (4), 329–344 (2008).

12. I. V. Oseledets, D. V. Savostyanov, and E. E. Tyrtyshnikov, "Cross Approximation in Tensor Electron Density Computations," Numer. Linear Algebra Appl. **17** (6), 935–952 (2010).