РАСПАРАЛЛЕЛИВАНИЕ ИТЕРАЦИОННОГО АЛГОРИТМА УМЕНЬШЕНИЯ ШИРИНЫ ЛЕНТЫ РАЗРЕЖЕННОЙ МАТРИЦЫ

Н. В. Старостин, В. А. Заиграев

Институт информационных технологий механики и математики
ННГУ им. Н. И. Лобачевского, г. Нижний Новгород

В работе рассматривается известная и актуальная задача минимизации ширины ленты разреженной матрицы [1]. Данная задача, в частности, возникает на этапах подготовки разреженных структур (матриц, аппроксимационных сеток, графов) в процессах физико-математического моделирования на Супер-ЭВМ. Размерности практических задач исчисляются десятками и сотнями миллионов строк/столбцов матриц. Уменьшение локальных ширин разреженных структур позволяет сэкономить вычислительные ресурсы, такие как: память под хранение данных, время выполнения расчетов, пересылки данных в коммуникационных средах.

В качестве исходных данных в задаче уменьшения ширины ленты выступает квадратная разреженная симметрическая матрица A. Локальной шириной называют величину, которая соответствует расстоянию между главной диагональю и крайним ненулевым элементом *i*-ой строки $\beta_i(A) = \max_{j=\overline{1,n}} \left\{ |i-j| \middle| a_{ij} \neq 0 \right\}$. А шириной ленты матрицы называют локальную ширину ленты, максимальную по значению $\beta(A) = \max_{i=\overline{1,n}} \beta_i(A)$

Решением задачи является перестановка (матрица перестановок) P, согласно которой перенумерация строк/столбцов исходной матрицы обеспечивает максимальное сужение ширины ленты $F(P) = \beta\left(PAP^T\right) \to \min$.

По нескольким причинам, для решения этой задачи применяются эвристические алгоритмы. Главная причина – это то, что проблема уменьшения ширины ленты относится к классу NP-трудных проблем [2], то есть для нее не существует алгоритма, решающего ее за полиномиальное время. Также, к использованию эвристических алгоритмов подталкивает большая размерность современных задач, делающая применение прямых алгоритмов неэффективным по объему затрат вычислительных ресурсов: памяти и процессорного времени.

Для практического применения важно, чтобы общее время решения задачи уменьшилось, то есть суммарное время работы алгоритма уменьшения ширины ленты и решения эквивалентной задачи с переупорядоченой матрицей было меньше, чем время решения задачи с исходной матрицей. Поэтому на практике применяются быстрые конструктивные алгоритмы, например, алгоритм Катхилла-Макки, Кинга, Слоана. Общей проблемой названных алгоритмов можно назвать то, что они не позволяют улучшить найденное решение. Задачей этой работы было разработать такой параллельный итерационный алгоритм, который, получая на вход начальное упорядочение трансформирует его так, чтобы значение критерия улучшилось по сравнению с исходным. Один из плюсов выбора итерационного алгоритма является возможность прекратить его работу в любой момент времени с получением определенного результата, до срабатывания условия останова.

В качестве операции трансформации был выбран обмен пары соседних строк и столбцов, обеспечивающий сужение ленточный структуры матрицы.

В целях распараллеливания алгоритма было решено разбить перестановку на отдельные пересекающиеся фрагменты, каждый из которых обрабатывается независимо. Размер каждого фрагмента должен превышать начальную ширину ленты матрицы. Это позволяет гарантировать то, что изменения в текущем фрагменте оказывают влияние только на локальные ленты соседних фрагментов. Пересечение фрагментов необходимо для сохранения возможности перехода вершин входящих в состав одного фрагмента в другой. Для минимизации числа проверок и улучшения скорости работы, была также введена оценка перспективности отдельных фрагментов – под перспективностью понимается потенциальная возможность улучшения локальных лент в рамках текущего фрагмента. Изначально принимаются все фрагменты перспективными. Если после обработки не было совершено ни одного обмена, то фрагмент помечается как неперспективный и далее не обрабатывается. Когда в некотором фрагменте происходит обмен, соседние к нему и он сам помечаются как перспективные. При обработке фрагментов возможно отсортировать их в порядке убывания перспективности, что может быть полезно для экономии вычислительных ресурсов.

Параллельная схема работы алгоритма реализуется методом разбиения всех фрагментов на четные и нечетные в порядке их положения в перестановке. Каждая итерация алгоритма состоит из двух тактов. На первом такте в параллельном режиме обрабатываются нечетные фрагменты, на втором такте – четные фрагменты. Поскольку размер фрагмента гарантирует, что локальные изменения в нем влияют лишь

Результаты запусков алгоритма

Название	N	Заполнение	RCM	Iterative		Parallel	
				Лента	Время, мс	Лента	Время, мс
lshp_265	265	1009	19	19	22	19	71
eris1176	1176	9864	183	183	1074	104	1261
bcspwr09	1723	6511	170	80	9413	81	2144
bcspwr10	5300	21842	490	490	17297	179	10065
hvdc1	24842	158426	4251	4251	26103	3295	3528
bcsstk30	28924	1036208	5040	5040	1635	4605	2054
TSOPF_FS_b162_c4	40798	2398220	40287	40207	22210	40218	13687
bcsstk32	44609	1029655	4556	4556	1635	4524	1108
TSOPF_FS_b39_c19	76216	1977600	76067	75866	57682	75832	12167
LeGresley_87936	87936	280523	4734	4734	2753	3159	3945
hvdc2	189860	1339638	3513	3513	27722	3161	4557

на соседние к нему вершины, в рамках каждого такта обработка всех фрагментов происходит полностью независимо в параллельном режиме.

С учетом распараллеливания работы алгоритма и при ширине ленты меньшей величины $\frac{N}{2\rho}$, ускорение его работы линейно зависит от p – числа процессоров.

Представленный алгоритм был реализован на базе платформы Java SE 9 и протестирован на ПК следующей конфигурации: IntelCoreI7-3520M/8 ГГб ОЗУ. Для тестирования были взяты матрицы различных размерностей (от 180 до 189860) из открытых источников [3]. Начальное упорядочение для каждого запуска генерировалось с помощью конструктивного алгоритма Катхилла-Макки [4]. Затем полученная перестановка и исходная матрица подавались на вход последовательной итерационной схемы, а также параллельной версии данной схемы, работающей в четыре вычислительных потока. Запуски показали, что качество работы последовательной и параллельной схемы сравнимы и позволяют получать улучшение ширины ленты в среднем до 21 % по сравнению с упорядочением Катхилла-Макки. Ускорение параллельной версии соответствует теоретической оценке, и на рассматриваемом наборе задач максимальное время работы составило 14 секунд. Результаты работы алгоритма представлены в таблице.

В дальнейшем, полученный алгоритм будет развит в сторону увеличения размерностей обрабатываемых матриц. В рамках этого развития предполагается переход в сторону многоуровневого алгоритма [5], то есть введения дополнительных этапов, последовательность которых позволит улучшить качество решения и время работы алгоритма. Также перспективно развитие в сторону распределенного алгоритма, который обеспечивает упорядочение разреженных структур практически значимых порядков.

Литература

1. Писсанецки С. Технология разреженных матриц. Пер. с англ. М.: Мир, 1988;

2. Paradimitriou Ch. H. The NP-Completeness of the bandwidth minimization problem // Computing. 1976 Vol. 16P. 263-270;

3. The University of Florida Sparse Matrix Collectionhttps://sparse.tamu.edu/;

4. Cuthill E., McKee J. Reducing the bandwidth of sparse symmetric matrices // Proc. 24th Conference of the ACM, Brandon Press, New Jersey, P. 157 – 172 1969;

5. Старостин Н. В., Панкратова М. А. Многоуровневые алгоритмы декомпозиции графа данных для параллельных вычислений на гетерогенной вычислительной системе. Вопросы атомной науки и техники. Серия: Математическое моделирование физических процессов. 2016. Т. 1. С. 60–68.