УДК 004.021

ОБ ОДНОМ ПОДХОДЕ К СРАВНЕНИЮ МАСШТАБИРУЕМОСТИ ПАРАЛЛЕЛЬНЫХ ПРОГРАММ

А. М. Теплов[1]

Предложен подход к анализу масштабируемости параллельных приложений. Введено понятие масштабируемости, учитывающее влияние на качество выполнения программ нескольких параметров запуска. Введено и обосновано понятие метрики масштабируемости и оценки масштабируемости параллельной программы. Описан также подход к сравнению масштабируемости параллельных программ, основанный на использовании оценки масштабируемости, входящей в метрику масштабируемости параллельной программы. Приведены результаты исследования масштабируемости, оценки и метрики масштабируемости и результаты сравнения масштабируемости нескольких параллельных программ.

Ключевые слова: масштабируемость, параллельные вычисления, исследование масштабируемости, сравнение масштабируемости, метрика масштабируемости, оценка масштабируемости, эффективность параллельных программ.

1. Введение. Важнейшим критерием работы любого параллельного приложения в области высокопроизводительных вычислений является время его выполнения. На время выполнения влияет множество факторов, которые могут быть как не зависящими от самого приложения, так и связанными с работой приложения или с параметрами его запуска. Для оценки качества работы параллельных программ применяют различные динамические характеристики, такие как ускорение, вычислительная сложность задачи, эффективность реализации и распараллеливания, стоимость вычислений и др. [1]. Динамические характеристики позволяют более детально оценить качество работы исследуемого параллельного приложения и понять, насколько эффективно используются при этом вычислительные ресурсы [1, 4]. На их реальные значения оказывают влияние те же факторы, что и на время выполнения всего приложения. Из этих факторов можно выделить в первую очередь параметры запуска параллельного приложения, так как их мы можем менять при разных прогонах программы.

В настоящей статье рассмотрен вопрос анализа обобщающей характеристики приложения — масштабируемости и предложен подход к сравнению масштабируемости разных параллельных приложений.

2. Масштабируемость параллельных программ. Понятие масштабируемости параллельной программы может быть определено по-разному. В различных источниках [3–7] определения масштабируемости могут довольно значительно отличаться друг от друга, отражая тот факт, что понятие масштабируемости сложное и требует тщательного анализа. Перспективным является подход к анализу масштабируемости параллельного приложения как зависимости показателей качества выполнения приложения от нескольких *параметров запуска* [8] параллельного приложения. Кроме того, в рамках такого анализа важно рассматривать различные связанные *динамические характеристики* [8]. Такой подход подразумевает рассмотрение масштабируемости как многомерной функции от большого числа параметров запуска.

Масштабируемостью [8] параллельного приложения будем называть свойство параллельной программы, характеризующее зависимость изменения динамических характеристик ее работы от изменения параметров запуска этого приложения. При анализе и оптимизации параллельных программ часто появляется необходимость сравнивать масштабируемость различных программ.

Конфигурацией запуска будем называть набор значений $K = (a, b, c, \ldots)$ параметров запуска параллельного приложения (например, размер задачи — a, число процессов — b, размер блока — c, число процессов на узел и др.).

Масштабируемость параллельной программы характеризует, каким образом меняются динамические характеристики параллельной программы при использовании различных конфигураций запуска параллельной программы.

Область рассматриваемых значений параметров представляет собой набор всех возможных значений параметров, входящих в конфигурацию запуска параллельного приложения.

[1] Московский государственный университет им. М. В. Ломоносова, Научно-исследовательский вычислительный центр, Ленинские горы, 119992, Москва; мл. науч. сотр., e-mail: alex-teplov@yandex.ru

Так как одно и то же приложение на разных областях рассматриваемых значений параметров запуска может иметь существенно отличающиеся свойства масштабируемости, то при рассмотрении масштабируемости параллельного приложения важно указывать, на какой области значений параметров проведено исследование.

Как правило, при анализе масштабируемости параллельного приложения рассматривают масштабируемость при изменении одного параметра, реже (например, при рассмотрении функции изоэффективности [14]) рассматривают два параметра. Для анализа масштабируемости приложений обычно используют одну и ту же характеристику параллельной программы. Чаще всего используют ускорение, производительность или эффективность, но может использоваться и представлять большой интерес любая другая динамическая характеристика, которая является показателем качества выполнения параллельной программы. Однако такое представление зависимости (например, ускорения от числа процессоров) вызывает сразу множество вопросов и уточняющих замечаний о параметрах проведения исследования. Вопросы вызывает понимание того, что на вид полученной зависимости влияют разные параметры запуска приложения. Наличие такого влияния подразумевает и наличие зависимостей от этих параметров. Такие зависимости также рассматриваются при анализе приложений [9].

По типу изменяемого параметра запуска приложения в литературе различают сильную масштабируемость, масштабируемость вширь и слабую масштабируемость [10]. Можно было бы аналогично выделить и другие типы масштабируемости. Набор изменяемых параметров в конфигурации запуска определяет, в каких координатах мы будем рассматривать динамические характеристики приложения. Чаще всего такими параметрами являются число использованных процессов и размерность решаемой задачи.

Параметров может быть больше одного, поэтому двумерное представление является частным случаем представления этой зависимости. Если рассматривать изменение производительности параллельного приложения от двух параметров конфигурации запуска: числа процессов и размерности решаемой задачи, то тогда график масштабируемости приобретает вид трехмерной поверхности.

В рамках исследования важно рассматривать несколько динамических характеристик параллельной программы. К таким характеристикам, помимо традиционных, таких как ускорение, эффективность и производительность, следует относить и важные данные показателей загруженности отдельных компонентов вычислительной системы. Эти данные могут быть получены из систем мониторинга и отчетов о выполнении задач, таких как Job Digest [11]. Использование таких данных позволяет делать выводы о причинах найденных особенностей в полученных зависимостях [8, 12]. Найденные особенности изменения различных показателей качества выполнения могут быть сложно объяснимы, если рассматривать каждую зависимость отдельно, и потому интерпретация полученных результатов экспериментов будет затруднена. Однако, рассматривая комплексно зависимость значений различных динамических характеристик от разных параметров запуска в одних и тех же координатах, можно находить корреляции в поведении таких функций, которые обусловлены взаимным влиянием этих показателей.

На рис. 1 изображен график масштабируемости блочного перемножения матриц. Из конфигурации запуска для анализа выбраны два параметра: число процессов и размер матрицы. В качестве исследуемой динамической характеристики использована производительность приложения.

Как видно из этого графика, по обеим осям получена достаточно гладкая зависимость производительности, т.е. небольшие изменения параметров запуска соответствуют небольшим изменениям в производительности. Наличие области на графике с высокой производительностью свидетельствует о наличии особенности исследуемой реализации, связанной, вероятно, с влиянием сверхмасштабируемости, вызванной удачным размещением данных в кэш-памяти.

3. Метрика масштабируемости параллельного приложения. При анализе масштабируемости нескольких приложений встает вопрос поиска критерия качества, по которому можно было бы сравнивать их масштабируемость. Иными словами, необходима метрика масштабируемости, которая позволяла бы характеризовать масштабируемость исследуемого приложения так, чтобы она не зависела от особенностей запуска приложений и позволяла сравнивать приложения, границы областей значений параметров запуска которых могли бы даже не пересекаться или же не иметь общих значений.

Целью исследования при изучении масштабируемости параллельного приложения, как правило, является поиск разумных пределов параметров запуска приложения, в которых эффективность остается на достаточно высоком уровне. Чаще всего рассматривают пределы для числа использованных процессов. При увеличении этого параметра и размера решаемой задачи эффективность имеет свойство уменьшаться из-за роста накладных расходов. С этим связано также уменьшение положительного эффекта от снижения вычислительной сложности на один процесс.

Поэтому для получения метрики масштабируемости в качестве исследуемой динамической харак-

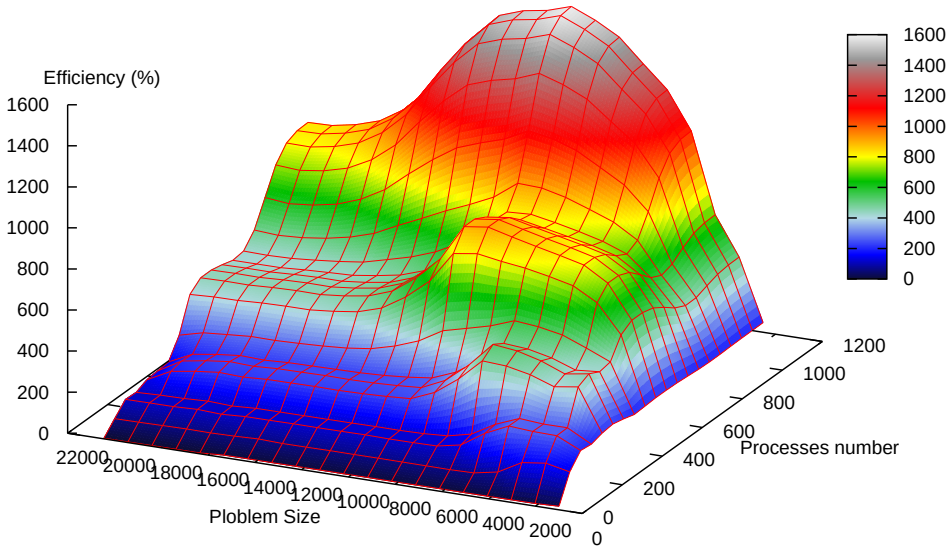

Рис. 1. Зависимость производительности блочного перемножения матриц от числа процессов и размера задачи

теристики при изменении параметров запуска будем использовать эффективность. Увеличение эффективности будет оказывать положительное влияние на качество выполнения программы, а уменьшение — отрицательное. Интенсивность изменения тоже играет важную роль: приложения могут изменять эффективность в определенных пределах, но либо резкими скачками, либо плавно и стабильно.

Все вышеописанные параметры и результаты проведенного исследования должны входить в искомую метрику масштабируемости параллельного приложения.

4. Вывод оценки масштабируемости. Формулировка подхода к сравнению. Предполагая, что о параллельном приложении собрана вся информация, характеризующая его работу на пространстве рассматриваемых значений параметров запуска, можно ввести метрику масштабируемости приложения. Кроме того, предположим, что собранные данные представимы в виде прямоугольной сетки.

Рассмотрим выборку результатов экспериментов, в которой берем максимальную эффективность работы по всем проведенным запускам, сгруппированным по одинаковому числу процессов и размеру решаемой задачи.

На рис. 2 показано, как может выглядеть трехмерное графическое представление такой выборки для приложения, реализующего скалярное произведение векторов.

Для расчета метрики рассмотрим выборку результатов экспериментов, сгруппированных по одинаковому числу процессов P_n и по размеру решаемой задачи D_n с максимальной эффективностью выполнения программы по всем проведенным запускам.

Результаты эксперимента соберем в группы по четыре значения эффективности из соседних точек с близкими параметрами запуска так, чтобы такими группами покрыть всю рассматриваемую область параметров. Такие группы значений будем называть *элементами рассматриваемой области значений параметров*.

Тогда в рассматриваемой трехмерной сетке значений (по числу процессоров и размеру задачи) у каждой точки будет 3 соседних значения: для результата со значениями параметров (P_n, D_n) соседними будут результаты со значениями (P_{n+1}, D_n), (P_n, D_{n+1}), (P_{n+1}, D_{n+1}). Следовательно, три соседние точки будут у всех результатов, у которых значения параметров отличны от максимальных $P_n < \max(P)$ и $D_n < \max(D)$. Однако соседние результаты не будут найдены, если по каким-то причинам были пропущены какие-либо конфигурации запуска. В этом случае можно составить три соседние точки так, чтобы соседними стали те результаты, значения которых следуют за пропущенными. Таким образом, мы разо-

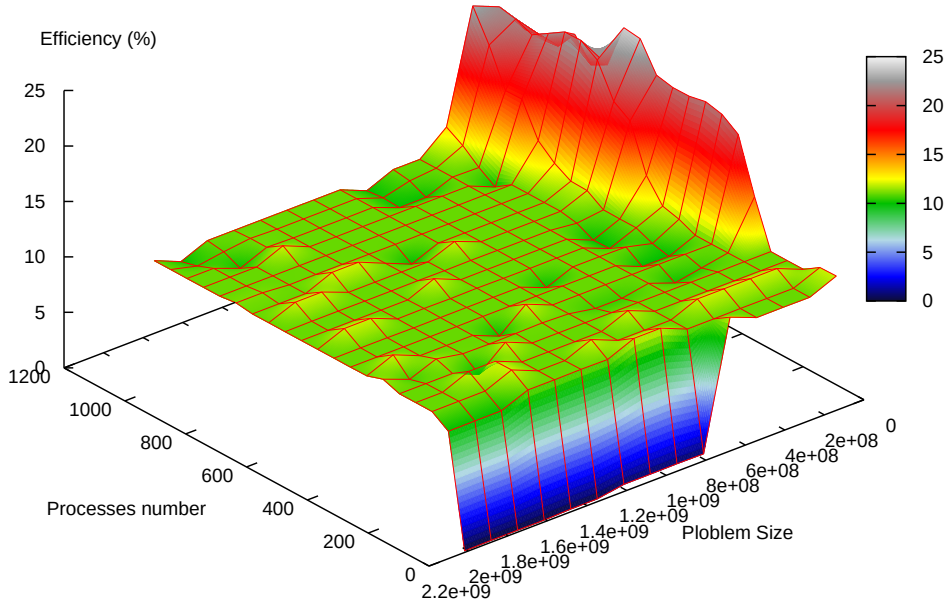

Рис. 2. Поверхность максимальной эффективности экспериментов, сгруппированных по равному числу процессов и размеру задачи

бьем рассматриваемую область значений параметров на отдельные элементы (см. рис. 3).

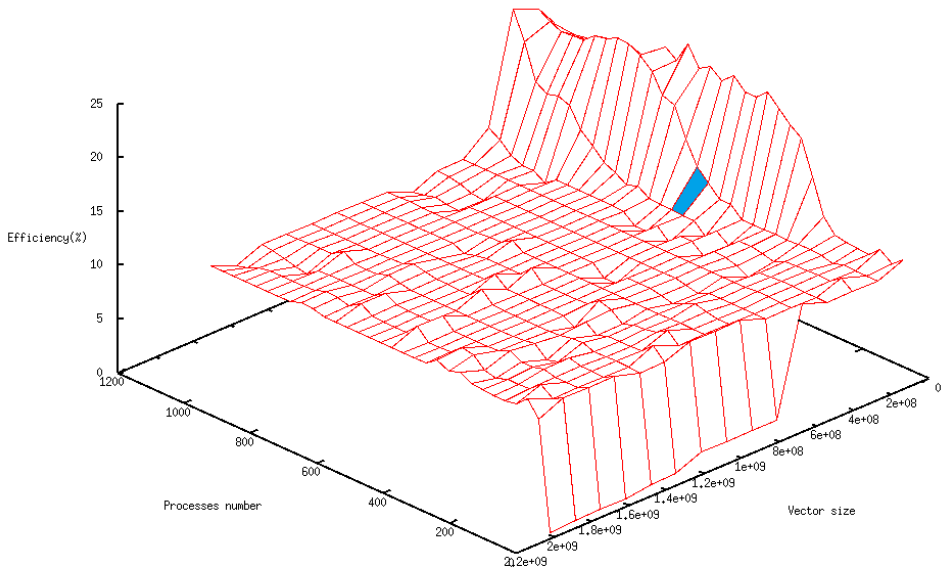

Рис. 3. Разбиение рассматриваемой области значений на элементы области. Синим цветом выделен один элемент рассматриваемой области

Для конфигурации с наименьшими значениями параметров запуска из области значений параметров запуска (P_n, D_n) можно найти ее соседние значения, так как в рассматриваемых результатах серии экспериментов изменяется число процессов и размер задачи. Далее, начиная с минимальной конфигурации, необходимо найти соседние значения для всех рассматриваемых результатов, для которых можно найти три соседние точки. Полученные группы результатов четырех экспериментов с соседними значениями будем рассматривать как вектор значений эффективности $(E_{11}, E_{12}, E_{21}, E_{22})$.

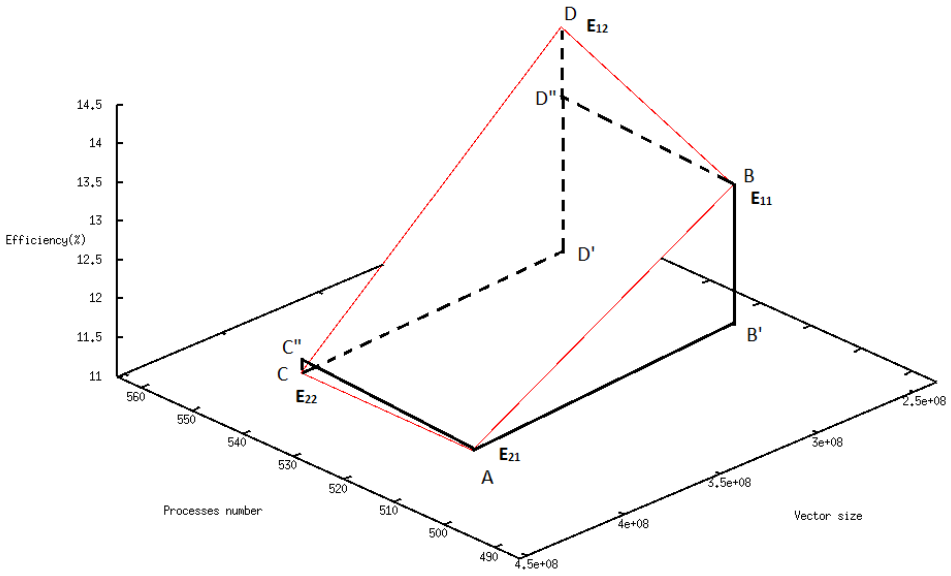

Рис. 4. Элемент рассматриваемой области значений параметров запуска

Для каждого такого элемента вектора вычислим приращения значения эффективности. Для этого рассмотрим изменение эффективности при увеличении значения каждого параметра. На рис. 4 показан один из элементов рассматриваемой области параметров запуска, где отражены также проекции отрезков, соединяющих эти точки. Тогда рассматриваемым приращениям соответствуют отрезки BB', DD', CC'', DD''.

Приращением эффективности по направлению оси X (увеличение числа процессов) будем называть среднее значение приращения $\Delta E_P = ((E_{11} - E_{12}) + (E_{21} - E_{22}))/2$, если E_{11} и E_{12} запущены с одним и тем же размером задачи, а число процессов у E_{12} следует за числом процессов у E_{11}. Аналогично для E_{21} и E_{22}. Вкладом элемента рассматриваемой области значений в общую оценку масштабируемости по числу процессов будем называть величину $markP = \Delta E_P * (p_2 - p_1)/(\max(p) - \min(p))$, где p_2 и p_1 — большее и меньшее значения параметра числа процессов на рассматриваемом элементе области параметров запуска.

Приращением эффективности по направлению оси Z (изменение размера задачи) будем называть среднее значение приращения $\Delta E_D = ((E_{11} - E_{21}) + (E_{12} - E_{22}))/2$, если E_{11} и E_{21} запущены с одним и тем же числом процессов, а размер задачи у E_{21} следует за размером задачи у E_{11}. Аналогично для E_{12} и E_{22}. Вкладом элемента рассматриваемой области значений в общую оценку по размеру задачи будем называть величину $markD = \Delta E_D * (d_2 - d_1)/(\max(d) - \min(d))$, где p_2 и p_1 — большее и меньшее значения параметра размера задачи на рассматриваемом элементе области параметров запуска.

Приращением эффективности по всем четырем значениям будем называть среднее значение приращения $\Delta E_A = (\Delta E_P + \Delta E_D)/2$, если E_{11} и E_{21} запущены с одним и тем же числом процессов, а размер задачи у E_{21} следует за размером задачи у E_{11}. Аналогично для E_{12} и E_{22}. Вкладом элемента рассматриваемой области значений в общую оценку будем называть величину $markA = \Delta E_A * ((p_2 - p_1) * (d_2 - d_1))/((\max(p) - \min(p)) * (\max(d) - \min(d)))$, где p_2 и p_1 — большее и меньшее значения параметра размера задачи на рассматриваемом элементе области параметров запуска.

Так как мы разбили всю рассматриваемую область значений параметров на элементы, то подобные

рассуждения применимы для каждого элемента, а всю область мы можем рассматривать как одномерный массив элементов рассматриваемой области.

Общей оценкой масштабируемости по числу процессов будем называть среднее значение вклада каждого элемента рассматриваемой области значений в общую оценку по числу процессов по всем элементам рассматриваемой области параметров запуска:

$$MarkProcs = (markP_1 + markP_2 + \ldots + markP_N)/N.$$

Общей оценкой масштабируемости по размеру решаемой задачи будем называть среднее значение вклада каждого элемента рассматриваемой области значений в общую оценку по размеру решаемой задачи по всем элементам рассматриваемой области параметров запуска:

$$MarkData = (markD_1 + markD_2 + \ldots + markD_N)/N.$$

Общей оценкой масштабируемости будем называть средний размер вклада каждого элемента рассматриваемой области значений в общую оценку по всем элементам рассматриваемой области параметров запуска:

$$MarkAll = (markA_1 + markA_2 + \ldots + markA_N)/N.$$

Тогда масштабируемость параллельного приложения на рассматриваемой области параметров запуска по размеру решаемой задачи и по числу процессоров будет характеризовать следующая *метрика*:

$$\min(p), \min(d), \max(p), \max(d), MarkProcs, MarkData, MarkAll, \max(E), \min(E),$$

где $\min(p)$, $\min(d)$, $\max(p)$, $\max(d)$ — границы значений параметров запуска рассматриваемой области; $MarkProcs$ характеризует интенсивность и направление изменения на рассматриваемой области по направлению увеличения числа процессоров; $MarkData$ характеризует интенсивность и направление изменения на рассматриваемой области по направлению увеличения размера задачи; $MarkAll$ характеризует интенсивность и направление изменения на рассматриваемой области по направлению увеличения всех параметров запуска; $\max(E)$, $\min(E)$ — границы изменения эффективности на рассмотренной области параметров запуска.

Из построения метрики вытекают следующие свойства:

— общая оценка масштабируемости для рассматриваемой области, состоящей из одного элемента, будет равна оценке для этого элемента;

— для двух приложений с подобными поверхностями изменения эффективности, но отличающимися между собой в каждом эксперименте на одну и ту же константу, оценка будет одинаковой, но отличаться будут максимальное и минимальное значения эффективности;

— для двух приложений, у одного из которых эффективность на всей рассматриваемой области значений является константой, а у второго меняется симметрично (до половины значений возрастает, после половины убывает с той же интенсивностью) оценка масштабируемости может совпасть и равняться нулю, но будут отличаться максимальное и минимальное значения эффективности; чтобы получить более детальную и различающуюся картину для таких приложений, целесообразно разбить рассматриваемую область на две и рассматривать их отдельно.

5. Результаты проведенных экспериментов. Обоснованность рассмотренного выше подхода была подтверждена на примере анализа масштабируемостей двух параллельных приложений: скалярное перемножение векторов и блочное перемножение матриц. При экспериментах использовались самые простые реализации без существенных оптимизаций, изменяющих классический алгоритм. Для распараллеливания использовалась библиотека Intel MPI 4.1.

Эксперименты проводились на суперкомпьютере “Ломоносов” [15] в НИВЦ МГУ. Для запусков использовались узлы с классическими процессорами Intel Xeon X5570 (два 4-ядерных процессора на узел), связанные между собой коммуникационной сетью QDR Infiniband 4x. Для компиляции использовался компилятор Intel 13.1.

Эксперименты на каждой конфигурации запуска производились по 3 раза с разным физическим расположением процессов по узлам. При каждом эксперименте приложение отрабатывало 4 раза. Из всех проведенных запусков выбирался запуск с наименьшим временем выполнения. Запуски проводились с размещением процессов по 8 на один узел.

5.1. Блочное перемножение матриц. Набор и границы значений изменяемых параметров запуска реализации алгоритма: число процессоров [4 : 1024]; размерность матрицы [1024 : 20 480] с шагом 1024.

В результате проведенных экспериментов был получен следующий диапазон эффективности реализации алгоритма: минимальная эффективность 4.71%; максимальная эффективность 31.72%.

Построим оценки масштабируемости выбранной реализации скалярного произведения векторов.

По числу процессов: -0.0436. При увеличении числа процессов эффективность убывает достаточно интенсивно на всей рассмотренной области изменений параметров запуска. Уменьшение эффективности на рассмотренной области работы связано с увеличением числа пересылок, с ростом числа процессов и, как следствие, ростом накладных расходов на организацию вычислений. Присутствует также область, на которой при увеличении числа процессов эффективность возрастает, но при дальнейшем росте опять снижается. Это объясняется декомпозицией данных, когда размер матрицы позволяет блокам укладываться в кэш-память. Кроме того, это подтверждает проявление того же явления, но со смещением по числу процессов, при увеличении вычислительной сложности задачи.

По размеру задачи: -0.0255. При увеличении размера задачи эффективность в целом уменьшается на рассматриваемой области, хотя и менее интенсивно, чем при увеличении числа процессов. Снижение эффективности объясняется тем, что при росте вычислительной сложности существенно возрастают объемы передаваемых данных. Кроме того, на всех рассмотренных размерах матрицы присутствует область возрастания эффективности. Это объясняется тем, что при малом размере задачи данные хорошо укладываются в кэш-память, что приводит к высокой эффективности работы приложения. При дальнейшем увеличении размера задачи эффективность уменьшается при выходе за границы кэш-памяти.

По двум направлениям: -0.000968. При рассмотрении увеличения как вычислительной сложности, так и числа процессов на всей рассмотренной области значений эффективность уменьшается. Интенсивность ее уменьшения не очень высока. В совокупности с тем фактом, что разница между максимальной и минимальной эффективностью на рассмотренной области значений параметров составляет почти 25%, это говорит о том, что уменьшение эффективности по всей области довольно равномерное, но интенсивно лишь на не очень больших по площади участках. На остальной области значений изменения эффективности не столь значительны и находятся на приблизительно одном и том же уровне.

На графике максимальной эффективности работы приложения (рис. 5) присутствуют особенности его масштабируемости, выявленные в выводах из оценки масштабируемости. Область с большим размером задачи и малым числом процессов соответствует конфигурациям запуска, на которых задача не помещалась полностью в память узла, потому эффективность выполнения на графике дополнена нулями.

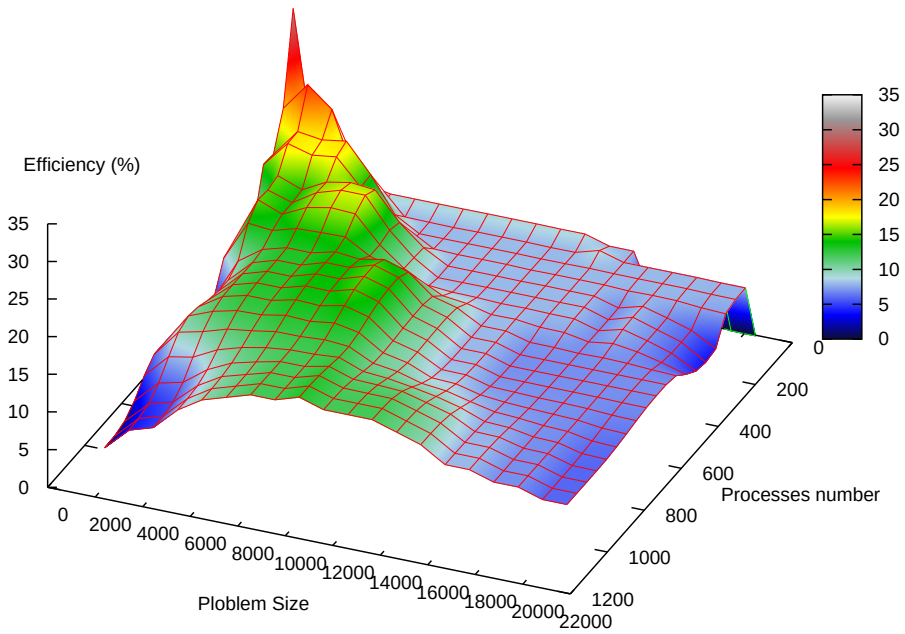

Рис. 5. График масштабируемости блочного перемножения матриц

5.2. Скалярное произведение векторов. Набор и границы значений изменяемых параметров

запуска реализации алгоритма: число процессоров [4 : 1024] с шагом 8; размер вектора [134 217 728 : 2 013 265 920] с шагом 134 217 728.

В результате проведенных экспериментов был получен следующий диапазон эффективности реализации алгоритма: минимальная эффективность 9.54 %; максимальная эффективность 24.52%.

Построим оценки масштабируемости выбранной реализации скалярного произведения векторов.

По числу процессов: 0.00414. При увеличении числа процессов эффективность увеличивается на рассмотренной области изменений параметров запуска, однако в целом увеличение не интенсивное. Увеличение эффективности на рассмотренной области работы параллельной программы объясняется тем, что при увеличении числа процессоров декомпозиция данных в какой-то момент приводит к тому, что данные лучше укладываются в кэш-память. Это подтверждает проявление этого явления, но со смещением по числу процессов, и при увеличении вычислительной сложности задачи.

По размеру задачи: -0.01385. При увеличении размера задачи эффективность в целом уменьшается на рассматриваемой области. Это объясняется тем, что при малом размере задачи данные хорошо укладываются в кэш-память, что приводит к высокой эффективности работы приложения при малом размере задачи. При увеличении размера эффективность уменьшается при выходе за границы кэш-памяти.

По двум направлениям: -0.000169. При рассмотрении увеличения как вычислительной сложности, так и числа процессов на всей рассмотренной области значений эффективность уменьшается, однако интенсивность ее уменьшения небольшая. Однако разница между максимальной и минимальной эффективностью на рассмотренной области значений параметров составляет почти 15%. Это говорит о том, что на поверхности присутствуют области с очень интенсивным изменением эффективности, но очень малые по площади. На остальной поверхности изменения эффективности незначительны и находятся приблизительно на одном и том же уровне.

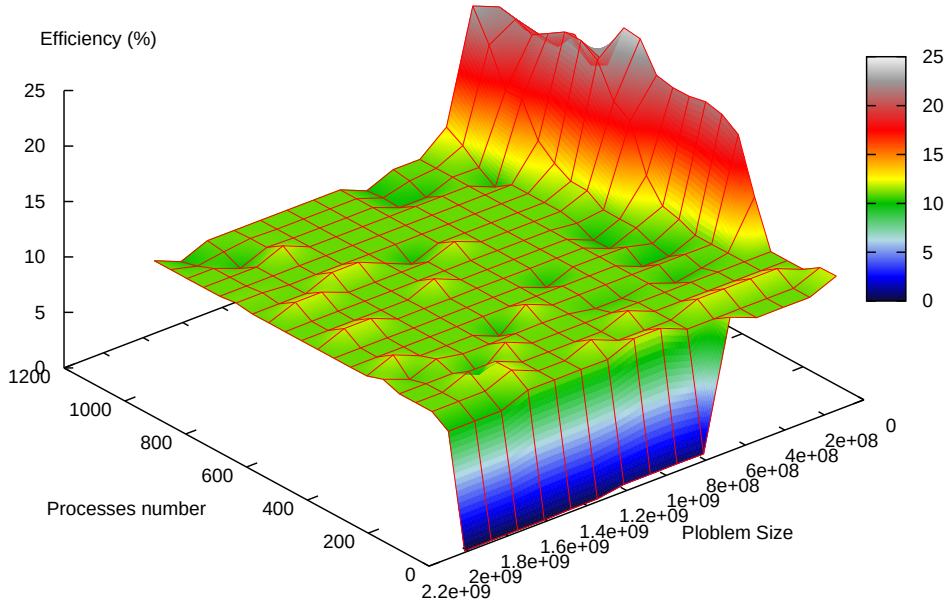

Рис. 6. График масштабируемости скалярного перемножения векторов

На графике максимальной эффективности работы приложения (рис. 6) присутствуют особенности масштабируемости, выявленные в выводах из оценки масштабируемости. Область с большим размером задачи и малым числом процессов соответствует конфигурациям запуска, на которых задача не помещалась полностью в память узла, поэтому на графике эффективность выполнения дополнена нулями.

5.3. Метод Холецкого. Набор и границы значений изменяемых параметров запуска реализации алгоритма: число процессоров [4 : 256] с шагом 4; размер матрицы [1024 : 5120].

В результате проведенных экспериментов был получен следующий диапазон эффективности реализации алгоритма: минимальная эффективность реализации 0.11%; максимальная эффективность реали-

зации 2.65%.

На рис. 7 приведен график изменения эффективности выбранной реализации разложения Холецкого в зависимости от параметров запуска.

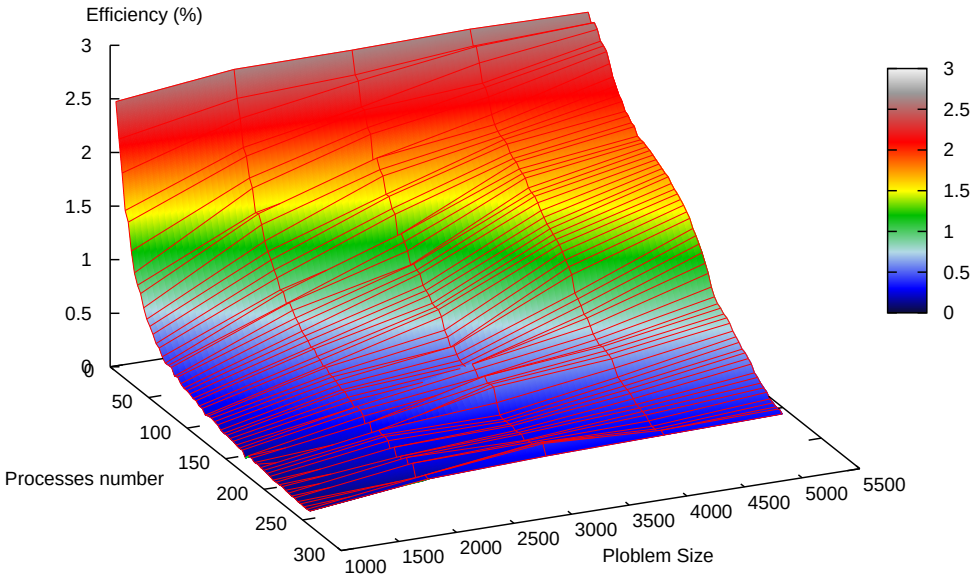

Рис. 7. Параллельная реализация метода Холецкого. Изменение эффективности в зависимости от числа процессоров и размера матрицы.

Построим оценки масштабируемости выбранной реализации разложения Холецкого.

По числу процессов: -0.000593. При увеличении числа процессов эффективность на рассмотренной области изменений параметров запуска уменьшается, однако в целом уменьшение не очень быстрое. Малая интенсивность изменения объясняется крайне низкой общей эффективностью работы приложения с максимумом в 2.65%, и значение эффективности на рассмотренной области значений быстро доходит до десятых долей процента (или уменьшается почти в 25 раз). Это свидетельствует о том, что на большей части области значений снижение эффективности интенсивно, хотя и численно само по себе мало. Это объясняется также тем, что с ростом вычислительной сложности падение эффективности становится не таким быстрым. Уменьшение эффективности на рассмотренной области работы параллельной программы объясняется быстрым ростом накладных расходов на организацию параллельного выполнения. С ростом вычислительной сложности задачи эффективность снижается так же быстро, но при больших значениях числа процессов. Это подтверждает предположение о том, что накладные расходы начинают значительно превалировать над вычислениями.

По размеру задачи: 0.06017. При увеличении размера задачи эффективность возрастает. Эффективность возрастает тем быстрее, чем большее число процессов используется для выполнения. Это подтверждает предположение о том, что размер задачи сильно влияет на эффективность выполнения приложения. Оценка показывает, что с ростом размера задачи эффективность на рассмотренной области значений параметров запуска сильно увеличивается. Кроме того, учитывая разницу максимальной и минимальной эффективности в 2.5% (или почти в 25 раз), можно сделать вывод, что рост эффективности при увеличении размера задачи наблюдается на большей части рассмотренной области значений.

По двум направлениям: 0.000403. При рассмотрении увеличения как вычислительной сложности, так и числа процессов на всей рассмотренной области значений параметров эффективность увеличивается, однако скорость увеличения эффективности небольшая. В совокупности с тем фактом, что разница между максимальной и минимальной эффективностью на рассмотренной области значений параметров небольшая, эффективность с увеличением масштабов возрастает, но медленно и с небольшими перепадами.

5.4. Linpack banchmark. Набор и границы значений изменяемых параметров запуска реализации

алгоритма: число процессоров [8 : 128] с шагом 8; размер матрицы [1000 : 100000] с шагом 1000.

В результате проведенных экспериментов был получен следующий диапазон эффективности реализации алгоритма: минимальная эффективность реализации 0.039%; максимальная эффективность реализации 86.7%.

На рис. 8 приведен график эффективности выполнения теста Linpack в зависимости от изменяемых параметров запуска.

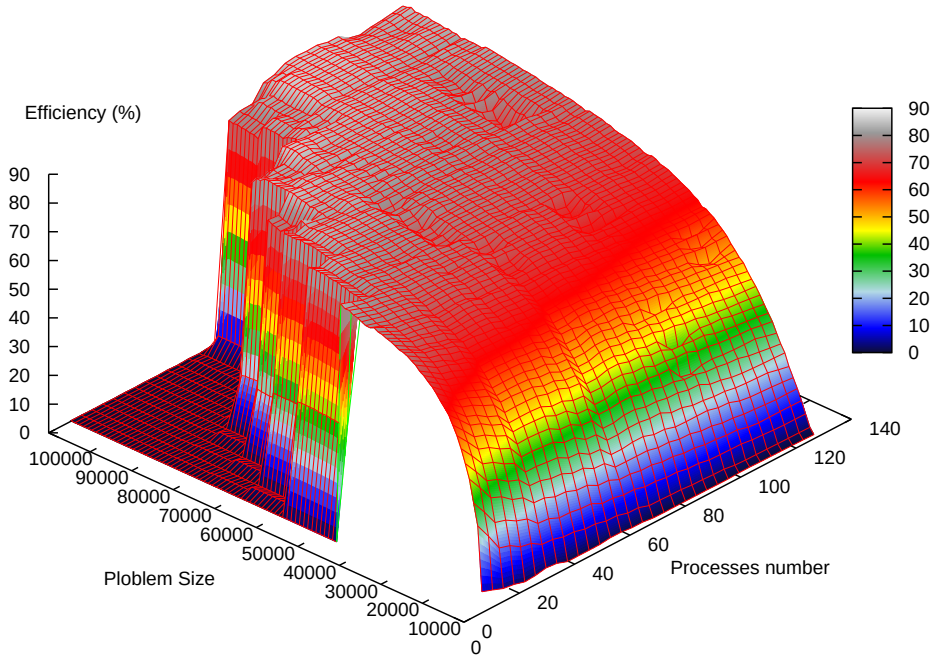

Рис. 8. Тест Linpack. Изменение эффективности в зависимости от числа процессоров и размера матрицы

Построим оценки масштабируемости теста Linpack.

По числу процессов: -0.061256. При увеличении числа процессов эффективность на рассмотренной области изменений параметров запуска уменьшается в целом довольно быстро. Большая интенсивность изменения объясняется высокой общей эффективностью работы при больших размерах задачи, однако при росте числа процессов эффективность равномерно уменьшается. Это объясняется также тем, что с ростом вычислительной сложности падение эффективности становится не таким быстрым. Уменьшение эффективности на рассмотренной области работы параллельной программы объясняется быстрым ростом накладных расходов на организацию параллельного выполнения. С ростом вычислительной сложности задачи эффективность снижается так же быстро, но при больших значениях числа процессов. Это подтверждает предположение о том, что накладные расходы начинают сильно превалировать над вычислениями.

По размеру задачи: 0.010134. При увеличении размера задачи эффективность возрастает достаточно быстро. Эффективность возрастает тем быстрее, чем большее число процессов используется для выполнения. Это подтверждает предположение о том, что размер задачи сильно влияет на эффективность выполнения приложения. Оценка показывает, что с ростом размера задачи эффективность на рассмотренной области значений параметров запуска сильно увеличивается.

По двум направлениям: 0.0000284. При рассмотрении увеличения как вычислительной сложности, так и числа процессов на всей рассмотренной области значений эффективность увеличивается, однако скорость увеличения эффективности не очень большая.

Метод Гаусса, прямой ход. Набор изменяемых параметров запуска реализации алгоритма и границы значений параметров алгоритма: число процессоров [4 : 256]; размер матрицы [1024 : 5120].

Эффективность выполнения реализации алгоритма: минимальная эффективность 0.11%; максимальная эффективность 6.65%.

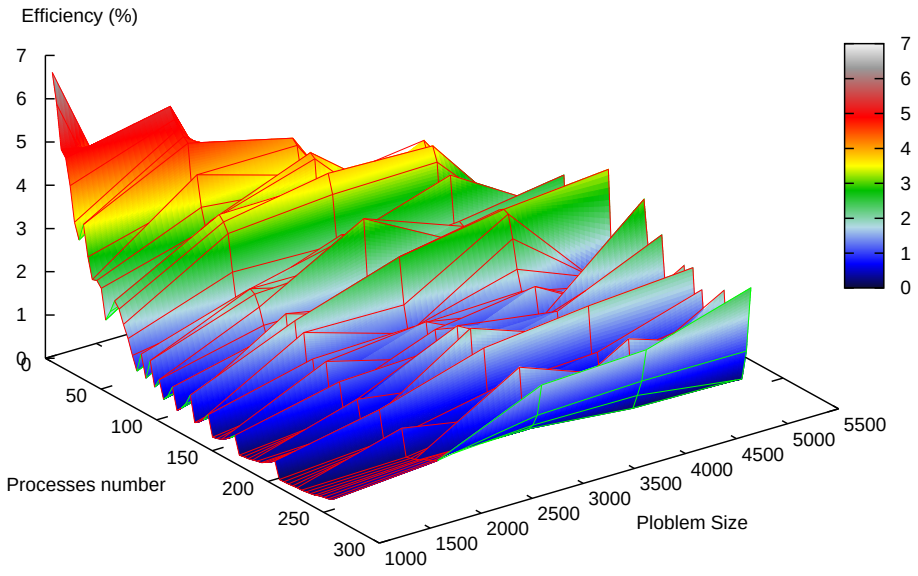

Рис. 9. Прямой ход метода Гаусса. Изменение производительности в зависимости от числа процессоров и размера матрицы.

В результате проведенных экспериментов был получен следующий диапазон эффективности реализации алгоритма.

По числу процессов: -0.000101. При увеличении числа процессов эффективность в целом уменьшается на рассматриваемой области, хотя и не очень интенсивно. Учитывая разницу между максимальным и минимальным значением эффективности в 6.5% (или почти в 60 раз) а также сильно "ломаную" структуру графика, можно сделать вывод, что на рассмотренной области снижение эффективности, скорее всего, достаточно равномерное. Сильные падения и увеличения эффективности выполнения компенсируют свои вклады в общую оценку и могут свидетельствовать о сильной чувствительности выбранной реализации к другим параметрам запуска, не учтенным в проведенном исследовании. Общее снижение эффективности объясняется тем, что при росте вычислительной сложности существенно возрастают объемы передаваемых данных. Могут присутствовать области возрастания эффективности на всех рассмотренных размерах матрицы. Это может объясняться увеличением вычислительной сложности задачи, что при постоянных накладных расходах на организацию параллельного взаимодействия приводит к общему увеличению эффективности работы.

По размеру задачи: -0.00674. При увеличении размера задачи эффективность в целом уменьшается на рассматриваемой области. Это может свидетельствовать о том, что при увеличении размера задачи возрастают и накладные расходы на организацию параллельного взаимодействия. Так как интенсивность снижения эффективности по этому направлению выше, чем по направлению числа процессов, скорее всего падение интенсивности при малом числе процессов более интенсивное, чем в присутствующих областях возрастания эффективности при большом числе процессов.

По двум направлениям: -6.621 e-05. При рассмотрении увеличения как вычислительной сложности, так и числа процессов на всей рассмотренной области значений эффективность уменьшается, однако интенсивность ее уменьшения очень мала. В совокупности с тем фактом, что разница между максимальной и минимальной эффективностью на рассмотренной области значений параметров составляет почти 6.5%, это говорит о том, что на поверхности присутствуют области с очень интенсивным изменением эффективности, но очень малые по площади. На остальной поверхности изменения эффективности незначительны и находятся на приблизительно одном и том же уровне.

6. Результаты сравнения масштабируемости. Если рассмотреть полученные оценки масштабируемости выбранных программ, то можно определить, масштабируемость какого приложения лучше

на рассмотренной области значений параметров. В проведенном исследовании принимало участие больше программ, чем описано выше. Кроме рассмотренных, исследовались также следующие приложения: метод сдваивания, параллельное суммирование элементов вектора, умножение матрицы на вектор. Все рассмотренные приложения использовались для разработки подхода к сравнению масштабируемости различных приложений в рамках проекта создания энциклопедии алгоритмов AlgoWiki [13], поэтому высокая эффективность работы не была приоритетом в выборе реализации приложения.

Выпишем в виде таблиц (см. табл. 1–3) сводные оценки масштабируемости, отсортировав их по возрастанию оценки.

Таблица 1. Сравнение оценок масштабируемости по числу процессов

Оценка	**Программа**
-0,061256	Linpack Benchmark
-0,0436	Перемножение матриц
-0,01517	Умножение матрицы на вектор
-0,00059	Метод Холецкого
-0,0001	Метод Гаусса, прямой ход
-3,06E-06	Схема сдваивания
-9,43E-07	Суммирование элементов вектора
0,00414	Скалярное произведение векторов

Таблица 2. Сравнение оценок масштабируемости по размеру задачи

Оценка	**Программа**
-0,0255	Перемножение матриц
-0,01385	Скалярное произведение векторов
-0,00674	Метод Гаусса, прямой ход
-0,0014	Умножение матрицы на вектор
6,43E-09	Схема сдваивания
1,88E-06	Суммирование элементов вектора
0,010134	Linpack Benchmark
0,06017	Метод Холецкого

Таблица 3. Сравнение общих оценок масштабируемости

Оценка	**Программа**
-0,00097	Перемножение матриц
-0,00017	Скалярное произведение векторов
-0,00015	Умножение матрицы на вектор
-6,62E-05	Метод Гаусса, прямой ход
-1,42E-07	Суммирование элементов вектора
-8,05E-08	Схема Сдваивания
2,84E-05	Linpack Benchmark
0,000403	Метод Холецкого

Полученные результаты сравнения оценок согласуются с визуальным сравнением графиков масштабируемости приложений. Низкая оценка теста Linpack в масштабируемости по числу процессов объясняется общей высокой производительностью программы, использующей библиотеку MKL. При малом размере задачи на рассмотренной области значений это приводит к сильному ухудшению качества работы

программы. Скалярное произведение векторов в силу декомпозиции данных при увеличении числа процессоров сильно увеличивает производительность работы, получая конфигурацию, при которой данные укладываются в кэш-память, когда проявляется эффект сверхмасштабируемости. Остальные программы на рассмотренных областях значений параметров запуска теряют эффективность выполнения не так интенсивно и потому находятся между этими приложениями.

При анализе оценок по размеру задачи и общей оценке приложения также качественно выстраиваются в иерархию, в которой они расположены в тех местах, которые отражают свойства полученной зависимости. Реализация метода Холецкого и тест Linpack сильно увеличивают эффективность выполнения при росте размера задачи. Однако тест Linpack быстрее приходит к насыщению своей эффективности, в то время как реализация метода Холецкого на рассмотренной области значений не доходит до насыщения и потому продолжает увеличивать эффективность на всей рассмотренной области значений. При перемножении матриц с ростом размера задачи падение эффективности наблюдается почти на всей рассмотренной области значений.

Промежуточное расположение приложений также согласуется с визуальным анализом графиков их масштабируемости.

Точность оценки близких по значениям оценки приложений, возможно, могла бы быть повышена путем замены размерности задачи на вычислительную сложность задачи.

7. Заключение. Описанный в настоящей статье подход к исследованию масштабируемости в ходе экспериментов показал, что он может быть успешно использован для анализа масштабируемости приложения при различных параметрах запуска. Использованная для оценки масштабируемости метрика отражает свойства масштабируемости исследованных приложений. Выведенная метрика масштабируемости позволяет интерпретировать с достаточной точностью результаты анализа масштабируемости. Результат сравнения оценок масштабируемости программ совпадает с результатами визуального сравнения графиков их масштабируемости. Полученная оценка масштабируемости позволяет сравнивать приложения на непересекающихся областях рассматриваемых значений параметров запуска. Точность оценки может быть повышена путем использования других численных методов и схем для вычисления оценок масштабируемости. Оценка масштабируемости по числу процессоров на данном этапе выглядит более информативной, потому что результат сравнения больше сходится с анализом данных о масштабируемости приложений. Дальнейшее повышение точности оценок масштабируемости позволит увеличить информативность предложенной метрики масштабируемости параллельной программы, и таким образом лучше отразить свойства исследуемого алгоритма.

Исследование выполнено в Московском государственном университете имени М. В. Ломоносова за счет гранта Российского научного фонда (проект № 14–11–00190).

СПИСОК ЛИТЕРАТУРЫ

1. *Воеводин В.В., Воеводин Вл.В.* Параллельные вычисления. СПб.: БХВ-Петербург, 2002.
2. *Антонов А.С., Жуматий С.А., Никитенко Д.А., Стефанов К.С., Теплов А.М., Швец П.А.* Исследование динамических характеристик потока задач суперкомпьютерной системы // Вычислительные методы и программирование. 2013. **14**. 104–108.
3. *Патил Р.В., Джордж Б.* Средства и приемы для выявления проблем параллельного выполнения // MSDN Magazine, 2008 (http://msdn.microsoft.com/ru-ru/magazine/cc546569.aspx).
4. *Левин М.П.* Параллельное программирование с использованием OpenMP. М.: Бином, 2008.
5. *Иванов Д.Е.* Масштабируемый параллельный генетический алгоритм построения идентифицирующих последовательностей для современных многоядерных вычислительных систем // Управляющие системы и машины. 2011. № 1. 25–32.
6. *Гергель В.П., Фурсов В.А.* Лекции по параллельным вычислениям. Самара: Изд-во Самар. гос. аэрокосм. ун-та, 2009.
7. *Назарова И.А.* Анализ масштабируемости параллельных алгоритмов численного решения задачи Коши // Наукові праці Донецького національного технічного університету. Серія: Інформатика, кібернетика та обчислювальна техніка. 2009. № 10. 21–26.
8. *Антонов А.С., Теплов А.М.* О практической сложности понятия масштабируемости параллельных программ // Высокопроизводительные параллельные вычисления на кластерных системах (HPC 2014): Материалы XIV Международной конференции. Пермь: Издательство Пермского национального исследовательского политехнического университета, 2014. 20–27.
9. *Позднеев А.В.* Исследование масштабируемости прямого метода решения уравнения Пуассона на вычислительной системе Blue Gene/P // Сб. тр. междун. суперкомп. конф. “Научный сервис в сети Интернет: суперкомпьютерные центры и задачи”. М.: Изд-во Моск. гос. ун-та, 2010. 123–132.

10. *Антонов А.С., Теплов А.М.* Исследование масштабируемости программ с использованием инструментов анализа параллельных приложений на примере модели атмосферы NH3D // Вестник Южно-Уральского государственного ун-та. Серия: вычислительная математика и информатика. Челябинск: Изд-во ЮУрГУ, 2013. **2**, № 1. 5–16.
11. *Адинец А.В., Брызгалов П.А., Воеводин Вад.В., Жуматий С.А., Никитенко Д.А., Стефанов К.С.* Job Digest — подход к исследованию динамических свойств задач на суперкомпьютерных системах // Вестн. Уфимского гос. авиационного технического ун-та. 2013. **17**, № 2. 131–137.
12. *Антонов А.С., Теплов А.М.* Использование данных системного мониторинга для определения факторов, уменьшающих масштабируемость приложения // Научный сервис в сети Интернет: многообразие суперкомпьютерных миров: Труды Всероссийской научной конференции (22–27 сентября 2014 г., г. Новороссийск). М.: Изд-во Моск. ун-та, 2014. 87–96.
13. *Воеводин Вл.В.* Суперкомпьютер "Ломоносов", прикладные пакеты и особенности алгоритмов // Труды первой межотраслевой научно-практической конференции "Суперкомпьютерные технологии в промышленности, 2014". СПб.: Крыловский гос. научный центр, 2014. 25–28.
14. *Grama A., Gupta A., Karypis G., Kumar V.* Introduction to parallel computing. Reading: Addison-Wesley, 2003.
15. *Воеводин Вл.В., Жуматий С.А., Соболев С.И., Антонов А.С., Брызгалов П.А., Никитенко Д.А., Стефанов К.С., Воеводин Вад.В.* Практика суперкомпьютера "Ломоносов" // Открытые системы. 2012. № 7. 36–39.

Поступила в редакцию
2.11.2014

An approach to the comparison of parallel program scalability

A. M. Teplov [1]

[1] *Lomonosov Moscow State University, Research Computing Center; Leninskie Gory, Moscow, 119992, Russia; Junior Scientist, e-mail: alex-teplov@yandex.ru*

Received November 2, 2014

Abstract: An approach to the scalability analysis of parallel programs is proposed. The introduced definition of scalability allows one to take into account the effect of startup parameters on the program execution quality. A concept of the scalability metric and scalability mark for parallel programs is also introduced. An approach to the comparison of parallel program scalability based on the use of the scalability mark as a part of the parallel program scalability metric is described. Some comparative results of scalability study using this approach are discussed by several examples of parallel programs.

Keywords: scalability, parallel computing, scalability analysis, comparison of scalability, scalability metrics, scalability mark, efficiency of parallel programs.

References

1. V. V. Voevodin and Vl. V. Voevodin, *Parallel Computing* (BKhV-Peterburg, St. Petersburg, 2002) [in Russian].

2. A. S. Antonov, S. A. Zhumatii, D. A. Nikitenko, et al., "Examination of Supercomputer System Jobs Flow Dynamic Characteristics," Vychisl. Metody Programm. **14**, 104–108 (2013).

3. R. V. Patil and B. George, "Tools and Techniques to Identify Concurrency Issues," MSDN Magazine (2008). http://msdn.microsoft.com/ru-ru/magazine/cc546569.aspx. Cited October 30, 2014.

4. M. P. Levin, *Parallel Programming Using OpenMP* (Binom, Moscow, 2008) [in Russian].

5. D. E. Ivanov, "A Scalable Parallel Genetic Algorithm to Construct Identifying Sequences for Modern Multicore Computers," Upravl. Syst. Mashin., No. 1, 25–32 (2011).

6. V. P. Gergel' and V. A. Fursov, Lectures on Parallel Computing (Samara Aerospace Univ., Samara, 2009) [in Russian].

7. I. A. Nazarova, "Scalability Analysis of Parallel Algorithms for Numerical Solution of Cauchy Problem," Vestn. Donetsk Nat. Tekh. Univ., Ser.: Inform. Kibern. Vychisl. Tekh., No. 10, 21–26 (2009).

8. A. S. Antonov and A. M. Teplov, "On Scalability Complexity of Parallel Programs," in *Proc. 14th Int. Conf. on High Performance Computing Using Cluster Systems, Perm, Russia, November 10–12, 2014* (Perm Nat. Res. Univ., Perm, 2014), pp. 20–27.

9. V. A. Pozdneev, "Scalability of the Direct Method for Solving Poisson's Equation on Blue Gene/P," in *Proc. Int. Conf. on Scientific Services & Internet: Supercomputing Centers and Applications, Abrau-Dyurso, Russia, September 20–25, 2010* (Mosk. Gos. Univ., Moscow, 2010), pp. 123–132.

10. A. S. Antonov and A. M. Teplov, "Application Scalability Study Using Tools to Analyze the Parallel Applications on the Example of the Atmosphere Model NH3D," Vestn. South Ural Univ., Ser.: Vychisl. Mat. Inform. **2** (1), 5–16 (2013).

11. A. V. Adinets, P. A. Bryzgalov, Vad. V. Voevodin, et al., "Job Digest: An Approach to Study Dynamic Properties of Tasks on Supercomputing Systems," Vestn. Ufimsk. Gos. Aviats. Tekhn. Univ. **17** (2), 131–137 (2013).

12. A. S. Antonov and A. M. Teplov, "Application of System Monitoring Data to Determine the Factors of Scalability Decrease," in *Proc. Int. Conf. on Scientific Services & Internet, Abrau-Dyurso, Russia, September 22–27, 2014* (Mosk. Gos. Univ., Moscow, 2014), pp. 87–96.

13. Vl. V. Voevodin, "The Lomonosov Supercomputer: Application Packages and Algorithmic Peculiarities," in *Proc. 1st Conf. on Supercomputing Technologies in Industry, St. Petersburg, Russia, October 17–18, 2014* (Krylov Gos. Nauch. Tsentr, St. Petersburg, 2014), pp. 25–28.

14. A. Grama, A. Gupta, G. Karypis, and V. Kumar, *Introduction to Parallel Computing* (Addison-Wesley, Reading, 2003).

15. Vl. V. Voevodin, S. A. Zhumatii, S. I. Sobolev, et al., "The Lomonosov Supercomputer in Practice," Otkrytye Sistemy, No. 7, 36–39 (2012).