Вычислительные эксперименты

Для вычислительных экспериментов был выбран класс липшицевых функций, моделируемых известным генератором GKLS [47]. Генератор тестовых задач GKLS порождает три класса тестовых функций: недифференцируемых, непрерывно дифференцируемых и дважды непрерывно дифференцируемых. Для каждого класса формируется 100 тестовых функций. В GKLS очень просто задать сложность генерируемых функций, определяя количество локальных минимумов, размеры областей притяжения и многое другое.

Тестирование алгоритма ДАМПД проводилось на наиболее сложном для реализации алгоритмов поиска глобального минимума, классе недифференцируемых функций. Для всех классов задач: число экстремумов равно 10; глобальный минимум равен -1; радиус притяжения глобального оптимума – 0,33. Эксперименты проводились на суперкомпьютерном кластере СГАУ «Сергей Королев». Кластер построен на базе линейки оборудования IBM BladeCenter с использованием блейд-серверов HS22 и обеспечивает пиковую производительность более 10 триллионов операций с плавающей точкой в секунду. Общее число процессоров/вычислительных ядер: 272/1184. Глобальный минимум вычислялся с точность  (по аргументам функции).

Первая версия алгоритма

Сначала был проверен базовый вариант двухфазного алгоритма глобальной оптимизации описанный в главе. Результаты первого эксперимента приведены в таблице 3.

Таблица 3 – Результаты эксперимента с базовой версией алгоритма ГО ММПД

|  |  |
| --- | --- |
| Время работы алгоритма, сек | 88,46 |
| Время накладных расходов, сек (% от общего времени) | 3,34 (3,75) |
| Число обращений к функции на этапе ГО (суммарно на всех процессорах) | 21152 (2937912) |
| Число обращений к функции на этапе ЛО (суммарно на всех процессорах) | 21095 (3024009) |

Таким образом, общее ускорение составило 141,121. На рисунках 12 и 13 приведены гистограммы, показывающие количество обращений к функции выполненное на каждом процессоре. Стоит отметить, что этап ЛО повторяется несколько раз, а гистограмма на рисунке показана только для одной итерации.

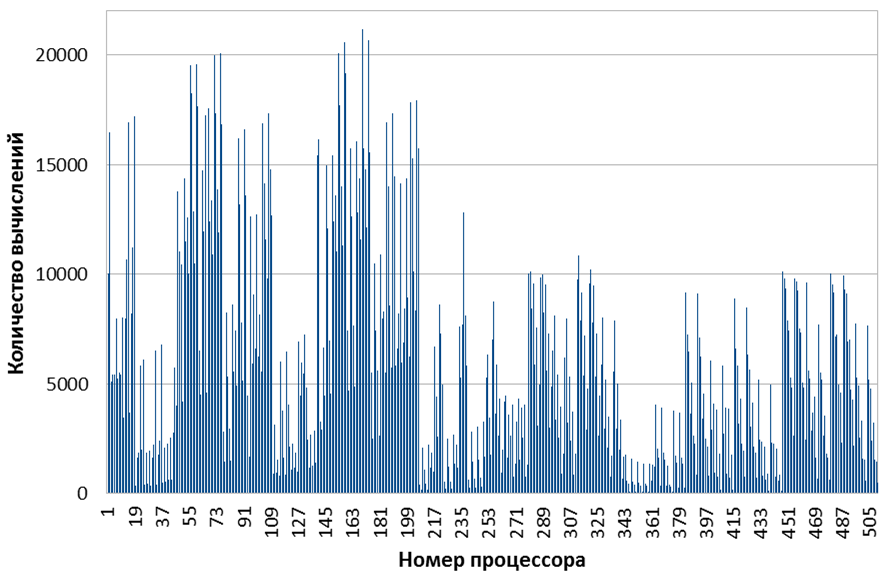


Рисунок 12 – Загрузка процессоров на этапе ГО в базовой версии

Из рисунков видно, что общая эффективность алгоритма невысока как на этапе ГО та и на этапе ЛО. Из-за прореживания на этапе ГО большая часть процессоров завершают деление своего начального параллелепипеда досрочно и вынуждены ждать завершения самого «нагруженного» процесса. Неравномерность распределения нагрузки на этапе ЛО обусловлена особенностью используемого алгоритма деформированных многогранников. Алгоритм сходится к локальному минимуму быстрее на более крутом участке и «долго ползет» на более пологом. Отсюда различное число вычислений функции на этапе ЛО.

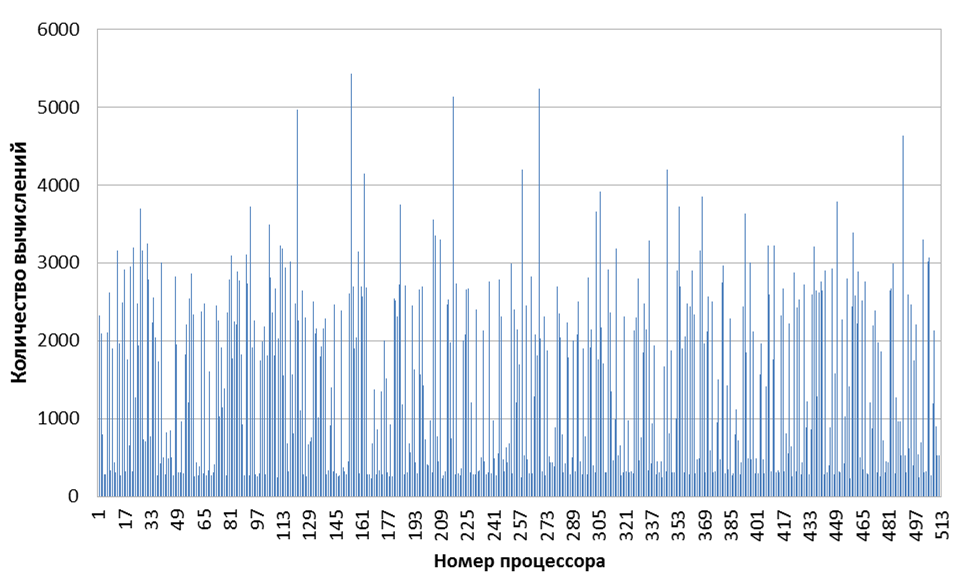


Рисунок 13 – Загрузка процессоров на этапе ЛО в базовой версии

Вторая версия алгоритма

Во второй версии параллельного алгоритма ГО с целью повышения эффективности вычислений, каждому из процессоров доступна информация о найденном текущем максимальном значении функции. Совместно с локальной оценкой константы Липшица, знание глобального значения максимума функции значительно сокращает количество проверяемых параллелепипедов по критерию прореживания, описанному в главе 3. С помощью технологии ГСП, модификация алгоритма из первой версии во вторую сводится к замене типа переменной, хранящей значение глобального минимума, с локальной на общую. Результаты второго эксперимента приведены в таблице 4.

Таблица 4 – Результаты эксперимента со второй модификацией алгоритма ГО ММПД

|  |  |
| --- | --- |
| Время работы алгоритма, сек | 82,34 |
| Время накладных расходов, сек (% от общего времени) | 3,54 (4,29) |
| Число обращений к функции на этапе ГО (суммарно на всех процессорах) | 16728 (840718) |
| Число обращений к функции на этапе ЛО (суммарно на всех процессорах) | 21443 (3081723) |

Таким образом, общее ускорение составило 102,76. Гистограмма распределение нагрузки на процессорах для этапа ГО показана на рисунке 14. По сравнению с первой версией алгоритма сократилось общее число вычислений, но при этом эффективность еще более уменьшилась. Общий вид гистограммы для ЛО не поменялся.

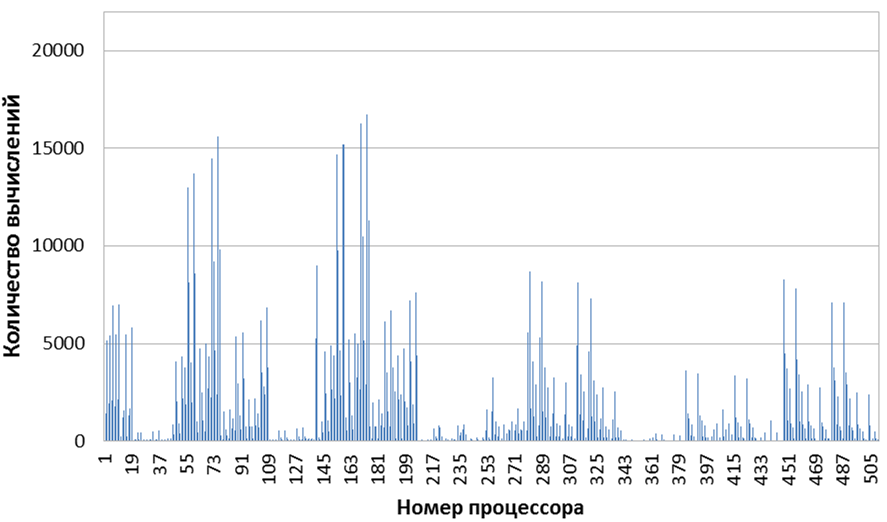


Рисунок 14 – Загрузка процессоров на этапе ГО во второй модификации алгоритма

Третья модификация алгоритма

Диаграммы распределения нагрузки на процессоры в эксперименте 2 говорят о еще меньшей рациональности алгоритма с точки зрения использования процессорного времени. Как и в первом случае это возникает в связи с неравномерностью прореживания параллелепипедов. В первой случае используется только 30% потенциальной вычислительной мощности, во втором случае менее 20%.

Для повышения эффективности алгоритма необходимо обеспечивать «быстрые» процессоры дополнительным заданием, пока остальные процессоры еще не закончили вычисления. Решением является переход от синхронной модели параллельных вычислений к асинхронной. Сначала попробуем асинхронную модификацию на этапе локального поиска.

Изначально готовится список локальных областей притяжения, больший, чем число процессоров. Все процессоры одновременно запускаются на поиск локального минимума каждый из своей точки. После завершения своего поиска ветвь-менеджер переходит в режим диспетчера и ждет сообщения о завершении от остальных ветвей. После приёма сообщения от любой параллельной ветви, ветвь-менеджер выдает новую точку отработавшей ветви и запускает её. Как только список локальных областей притяжения заканчивается, или проверено необходимое минимальное количество областей, ветвь-менеджер выходит из цикла приёма сообщений и все ветви просто завершают работу.

Для реализации данного подхода требуется либо написания более низкоуровневого актора, использующего функции MPI для ручного управления процессами, либо введение дуг синхронизации в модель алгоритма. Изначально модель параллельных вычислений в технологии ГСП является синхронной, но с помощью дуг синхронизации возможно моделирование асинхронной работы. Пример реализации для 4х процессоров с использованием дуг синхронизации показан на рисунке 15. Данная схема асинхронного управления часто называется менеджер-исполнитель, клиент-сервер и т.д.

Как показали эксперименты, применение асинхронного управления параллельными процессами позволяет значительно повысить ускорение и эффективность этапа поиска локального максимума. На рисунке видно более равномерное распределение вычислений по процессорам. Так эффективность этапа локального поиска возросла до 72%.

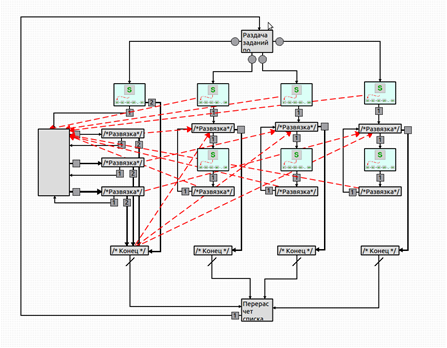


Рисунок 15 – Этап локальной оптимизации в режиме менеджер-исполнитель с синхронизацией

Хорошо показавший себя для этапа локального поиска асинхронный режим управления можно применить и для этапа глобального половинного деления. На этапе подготовки начального списка необходимо выполнить кратное 2n делений, больших, чем число процессоров. Затем с использованием той же схемы вычислений — менеджер-исполнитель — выполнять раздачу новых параллелепипедов уже закончившим работу процессам.

Результаты эксперимента с третьей версией алгоритма приведена в таблице 5.

Таблица 5 – Результаты эксперимента с третьей модификацией алгоритма ГО ММПД

|  |  |
| --- | --- |
| Время работы алгоритма, сек | 74,4 |
| Время накладных расходов, сек (% от общего времени) | 7,87 (10,57) |
| Число обращений к функции на этапе ГО (суммарно на всех процессорах) | 8170 (1949324) |
| Число обращений к функции на этапе ЛО (суммарно на всех процессорах) | 10607 (3199403) |

Таким образом, общее ускорение составило 274,204. На рисунках 16 и 17 приведены гистограммы, показывающие количество обращений к функции, выполненных на каждом процессоре. На рисунках мы видим более равномерное распределение нагрузки между процессорами. Здесь, в отличие от экспериментов 1 и 2, гистограмма для ЛО показывает общее число вызовов функции на каждом процессоре.

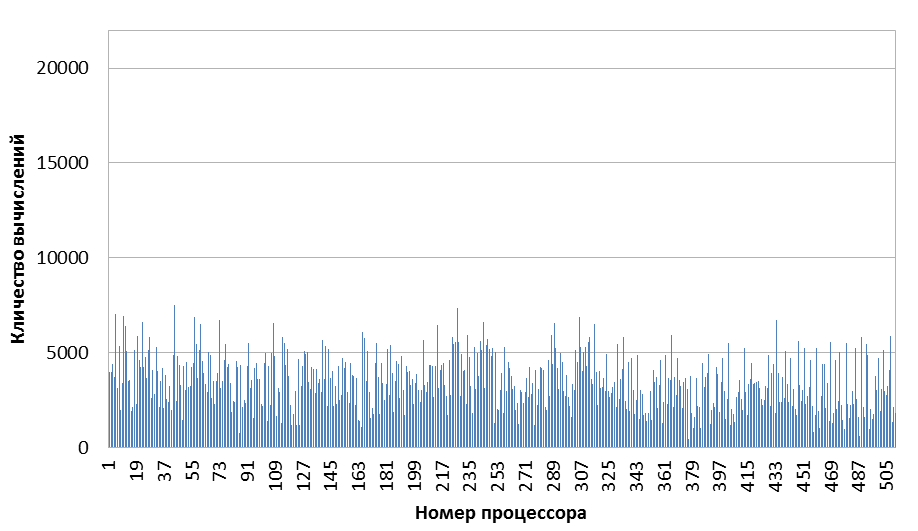


Рисунок 16 – Загрузка процессоров на этапе ГО в третьей модификации

Стоит отметить, что ощутимы результат для модели менеджер-исполнитель, как в последнем эксперименте, достигается для «тяжелых» функций, время вычисления которых много больше времени передачи параллелепипеда.



Рисунок 17 – Загрузка процессоров на этапе ЛО в третьей модификации

Рекорд

В качестве результата проделанных улучшений и настройки алгоритма приведем результаты эксперимента по нахождению глобального минимума для задачи с размерностью 15 той же тестовой функции. Практически это максимальная размерность, достигнутая для данной тестовой функции на суперкомпьютере. Параметры радиуса сходимости глобального максимума и количество локальных максимумов функции не поменялось. Оценка количества расчётов для размерности 15 при делении до параллелепипеда со стороной 0,25 дает число, порядка 2,15\*109. С учётом прореживания это число может уменьшиться[50], но ожидать, что оно уменьшится более чем на 2 порядка не стоит. При такой стороне параллелепипеда его диагональ получается 0,968, а следовательно радиус 0,484. Из этих оценок целесообразно взять LX=0,5.

На рисунке 18 показано как меняется радиус параллелепипеда от шага деления.

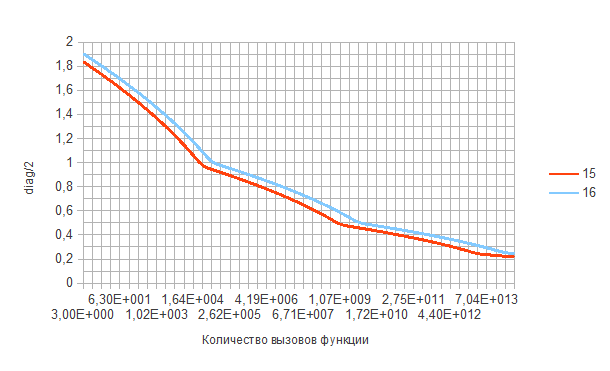


Рисунок 17 – Зависимость радиуса описанного вокруг параллелепипеда шара от сторон параллелипипеда

Отчёт о преодолении размерности 15 для задачи глобальной оптимизации. Условия эксперимента и параметры метода оптимизации приведены в таблице 16.

* Количество процессоров: 512
* Радиус зоны притяжения локального максимума: 0,2

|  |  |
| --- | --- |
| Параметры | 15 ND |
| lx | 0,5 |
| Номер и тип функции GKLS | 8 |
| Глобальный минимум найден | 4 |
| Общее число вызовов функций | 983770 |
| Общее ускорение | 286.183 |
| Ускорение глобального этапа | 310.346 |
| Ускорение локального этапа | 218.082 |
| Общее время работы алгоритма | 2413.7935 |