Равновесная упаковка объектов в заданной области по критерию минимизации длины связывающей сети

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 2](#_Toc505159901)

[1 ПОСТРОЕНИЕ МАТЕМАТИЧЕСКОЙ МОДЕЛИ 5](#_Toc505159902)

[1.1 Описание входящих данных 5](#_Toc505159903)

[1.2 Построение математической модели 6](#_Toc505159904)

[Выводы к разделу 1 12](#_Toc505159905)

[2 ОПИСАНИЕ МЕТОДА РЕШЕНИЯ 14](#_Toc505159906)

[2.1 Выбор и сравнение программных пакетов для решения задачи 14](#_Toc505159907)

[2.2 Описание метода поиска локального минимума 23](#_Toc505159908)

[2.3 Описание метода поиска глобального минимума 23](#_Toc505159909)

[Выводы к разделу 2 30](#_Toc505159910)

[3 ОПИСАНИЕ ПРОГРАММНОГО ПРОДУКТА 31](#_Toc505159911)

[3.1 Подготовка данных для решения с помощью IP-OPT 31](#_Toc505159912)

[3.2 Диаграмма вариантов использования 35](#_Toc505159913)

[3.3 Описание концепции объектно-ориентированного программирования при формировании модели программного продукта 38](#_Toc505159914)

[3.4 Сценарий работы пользователя с программным продуктом 48](#_Toc505159915)

[3.5 Результаты тестирования зависимости скорости от количества объектов 69](#_Toc505159916)

[Выводы к разделу 3 83](#_Toc505159917)

[ВЫВОДЫ 85](#_Toc505159918)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 86](#_Toc505159919)

# ВВЕДЕНИЕ

Представим, что некая фирма получила крупный заказ о поставке товара в заморское государство, следовательно удобным и дешевым транспортным средством является корабль. Изготовитель производит шаровой формы объекты (подшипники, ядра, мячи, апельсины, мотки ниток), это может быть что угодно. При погрузке судна необходимо учитывать небаланс системы в целом, иначе водоплавающее средство может пойти ко дну и нанести непоправимый ущерб финансовому положению и репутации компании.

Для решения этой задачи необходимо выяснить, насколько плотно можно уложить в пространстве большое количество одинаковых шаров. Если бы вместо шариков нужно было загрузить трюм судна одинаковыми кубиками, найти ответ было бы легко. Поскольку кубики плотно прилегают друг к другу и между ними не остаётся пустого места, они займут практически весь трюм (не считая небольших просветов вдоль стен и потолка). Однако шары нельзя упаковать так же плотно, как кубики: между ними всегда остаётся свободное место, и, если, несмотря на самую плотную упаковку, объём свободного пространства превысит четверть объёма трюма, его смело можно заполнить шариками и без риска выйти из гавани.

Возвращаемся к решению исходной задачи. Как выяснено, в задаче приходится говорить о постановке, которая должна учитывать несколько критериев размещения. Для производителя важно количество товара, которое он может переправить за один раз кораблем. В связи с этим, колеблется и стоимость единицы товара для заказчика и конечного покупателя. Исходя из этого, приходится учитывать критерий размещения объектов шаровой природы с учетом минимизации объема внешнего контейнера и ограничение по возможному отклонению от центра масс. Размещение будет производиться в некий «контейнер» шаровой формы, множество «объектов», должны быть упакованы в один или несколько контейнеров, а те, в свою очередь будут помещены в контейнер прямоугольной системы для устойчивости груза. Множество может содержать различные объекты с указанными пространственными характеристиками, которые могут быть различны, а могут быть одинаковы и повторяться для нескольких шаров.

Многие люди, потратив несколько минут на эксперименты с апельсинами или бильярдными шарами, приходят к ошибочному выводу, что проблема тривиальна. Уложим три шара на плоской поверхности так, чтобы их центры образовали равносторонний треугольник, а сами они касались друг друга. Затем будем подкладывать к ним другие шары так, чтобы каждый новый касался двух уже уложенных. Так получится первый слой шаров. Затем уложим второй слой, помещая каждый новый шар в «глубокую яму», или углубление, образовавшееся в центре каждой треугольной группы шаров первого слоя. В законченном виде второй слой не отличается от первого, он только сдвинут в горизонтальной плоскости. Если и следующие слои строить таким же способом, то получится так называемая гранецентрированная кубическая упаковка шаров, хорошо знакомая химикам и кристаллографам. При этом шары заполняют чуть больше 74% объёма пространства и каждому ясно, что уложить шары плотнее невозможно. Задачи упаковки относятся к классу NP-сложных.

К сожалению, математически не доказано, что это максимально достижимая плотность. Из полученных до сих пор верхних оценок плотности наименьшая была найдена в 1958 году К. А. Роджерсом из Бирмингемского университета; он доказал, что никакая упаковка шаров не может иметь плотность большую чем ~0,7796. Дело в том, что в доказательстве Роджерса не предлагается никакой упаковки шаров, плотность которой была бы близка к найденной оценке. Более того, сам Роджерс в статье, где сообщалось о полученном доказательстве, заметил: «многие математики верят, а все физики знают», что правильный ответ составляет около 74%. За четверть века, прошедшие с тех пор, ничего не изменилось, и задача о плотной упаковке шаров, такая простая на вид и столь трудная по существу, остаётся одной из важных нерешённых проблем в математике.

Рассмотренная выше задача отлично описывает суть поставленного вопроса, но рассмотрим задачу выше с некоторыми изменениями.

Заказчик транспортирует хрупкий товар в виде фарфоровых шаров разного радиуса, только теперь уже на автомобильном транспорте, который при плохом состоянии дорожного полотна может нести значительную разрушительную нагрузку на каждую единицу изделия. По-прежнему, необходимо вести учет состояния небаланса для равномерного распределения веса на каждую ось транспортного средства и во избежание механических разрушений в нашу область будут внесены дополнительные поролоновые элементы, которые будут смягчать удары о соседние предметы. Не мудрено, что «енергогасящие» элементы будут тоже, шарами, чтобы не внести разлад в структуру упаковываемых изделий, иначе придется пожертвовать дополнительным размером внешнего контейнера. Поролоновые шары, соответственно, будут разного радиуса, в силу необходимости прилегания к одному объекту больше, к другому меньше. Какие-то компонуемые объекты будут находиться ближе к поверхности, другие можно будет расположить снизу, но с большим промежуточным «смягчителем» – все это задача уравновешивания с учетом длины связи. Такая задача используется в космической технике при компоновке радиоэлементов на спутнике, в приборостроении, в задачах прокладки путей.

# 1 ПОСТРОЕНИЕ МАТЕМАТИЧЕСКОЙ МОДЕЛИ

## 1.1 Описание входящих данных

Сформируем исходные данные к дипломному проекту по теме «Равновесная упаковка объектов в заданной области по критерию минимизации длины связующей сети».

Исходным набором данных к указанной выше задачи являются: внешний контейнер, в который производится упаковка шаров, пространственные характеристики компонуемых объектов , симметричная двумерная матрица коэффициентов задающих количество связей между или дороговизну прокладки связующего звена между компонентами, тем самым указывая на степень важности близ расположения элементов в конечной системе, для решения задачи равновесной упаковки на ряду с критерием минимальной длины связи, необходимо указать или рассчитать по формуле веса каждого из шаров. Внешний контейнер зададим в форме шара, он обладает отличными от внутренних шаров пространственными характеристиками в евклидовом пространстве: он не имеет рассчитываемого или задаваемого веса и его центра всегда располагается в точке .

Желательными, но не обязательными данными для передачи на вход являются: программная точность вычислений (количество знаков после запятой, указывающих на прекращение вычислений), флаг вывода данных на экран и в текстовый файл (1-12) в зависимости от желаемой полноты получения отчетности.

Наиболее близко к окружающему нас пространству – трехмерное евклидово пространство, то есть, в нем выполнимы аксиомы евклидовой геометрии, соответственно, вся теория для решаемой задачи будет рассмотрена в евклидовом пространстве.

Необходимость указания некоторых входных данных в виде координат, радиуса шара, весов, матрицы связей дает, в дальнейшем, возможность произвести графическое отображение качества полученного решения.

Следующей группой параметров обязательных для первоначального указания являются ограничения системы в целом. Для каждого объекта размещения нужно выписать перечисленных ограничений, в совокупности это будет, система из всех ограничений, которая зависит от количества компонуемых шаров различного или одинакового радиуса. В нашей системе введены следующие ограничения: ограничение описывающее попадание шара в область внешнего контейнера, ограничение, исключающее попарное пересечение размещаемых шаровых объектов и ограничение небаланса системы.

Наборы данных задаются в виде чисел с плавающей запятой или как целочисленные в десятичной системе счисления.

## 1.2 Построение математической модели

Объект: это процесс решения задачи размещения объектов шаровой природы с учетом длины связывающей сети и состояния небаланса в заданной области

Предмет: моделирование и методы оптимизации компоновочных объектов сферической формы

Цель: Целью работы есть построение математической модели задачи минимизации длины связывающей сети между объектами с учетом баланса весов трехмерных объектов как задачи нелинейного программирования, обзор, выбор, решение поставленной задачи

Постановка задачи:

Для построения адекватных математических моделей математических задач в виде нелинейного программирования актуально аналитическое описание специальных ограничений с учетом минимально и максимально допустимых расстояний.

Описывается постановка и метод решения следующей задачи равновесной упаковки с учетом длины связующей сети.

Произведем декомпозицию основной задачи на две подзадачи для простоты понимания математической модели. Рассмотрим пространство , шар радиуса  с центром в точке – это внешний контейнер, совокупность шаров , радиусы  и массы  заданы пользователем предпочтительным ему способом. Упаковку шаров  в шаре назовем равновесной, если центр тяжести семейства шаров совпадает с центром шара [1].

Положение шаров  в пространстве задается параметрами размещения , , совпадающими с координатами их центров в . Зафиксируем параметры размещения  шаров , . Требуется найти такие параметры размещения , которые обеспечивают равновесную упаковку шаров   в шаре  минимального радиуса  [2]. Нетрудно видеть, что такая задача может рассматриваться как задача равновесной упаковки при описанных соответствующих ограничениях и условии небаланса [1,2].

Математическая постановка задачи имеет вид

 , (1)

при ограничениях:

, , (2)

, , , (3)

. (4)

Условие (2) описывает принадлежность шару  размещаемых шаров . Условие (3) задает условие попарного не пересечения размещаемых шаров между собой. Равенства (4) описывают условия равновесной упаковки [4].

Таким образом, имеем задачу математического программирования с  переменными ,, . В приведенной постановке радиусы  и массы  шаров являются константами. Зафиксируем , ,  и, положим, что радиусы упорядочены по возрастанию.

В соответствии с методом расширения пространства ослабим ограничения на радиусы шаров и будем считать их независимыми переменными.

Сформируем систему ограничений:

 (6)

 (7)

, (8)

где .

Система уравнений и неравенств (7)-(10) описывает множество всевозможных перестановок из чисел [5]. Таким образом, метод искусственного расширения пространства позволил сформировать задачу (1)-(8) в пространстве переменных ,

Существенным достоинством формализации задачи равновесной упаковки шаров в виде (1) - (8) является тот факт, что задача является квадратичной. Однако количество линейных ограничений в системе (5), (6) равно . Поэтому реализация классических методов нелинейной оптимизации ограничивается размерностью задачи. Вместе с тем учет свойств линейных и квадратичных функций на комбинаторных многогранниках позволяет в ряде случаев обходить возникающие трудности.

Задачи равновесной упаковки имеет широкое практическое приложение и достаточно широко исследуются в современной литературе. Заметим, что использование радиусов как независимых переменных в рамках иного концептуального подхода к решению некоторых классов задач упаковки рассмотрены в и подтвердили перспективность указанного направления [6].

Описанная выше задачи размещения геометрических объектов с учетом равновесия являются примерами задач, в которых цель оптимизации выполнение экстремальных требований к занятой части области размещения. В большинстве задач качество размещения определяют специальными свойствами, характеризующими размещенные объекты. Такие задачи возникают при конструировании электронных вычислительных систем, компоновке генеральных планов промышленных предприятий. Рассмотрим общую теорию этого класса задач на примере поиска минимальной длины связи между узлами (шарами) [7].

Пусть – множество объектов шаровой геометрической структуры. Обозначим через как матрицу связей. Каждый элемент этой матрицы равен количеству связей между , причем и для всех .

Необходимо разместить объекты в области таким образом, чтобы суммарная длина соединений, заданных матрицей С, была минимальной.

Как обычно, область свяжем с неподвижной системой координат, а размещаемые объекты – с собственными подвижными системами координат. Параметры размещения объекта обозначим через . Ясно, что описанная задача является задачей математического программирования. Функция цели в евклидовой метрике представима в следующем виде [8]:

Ограничения для такой системы представлены в виде неравенств (2)-(3), определяющих условия не пересечения размещаемых объектов и условия их размещения во внутренней области внешнего контейнера .

После рассмотренных выше задач в отдельности, произведем объединение постановок в единую. Новая целевая функция будет результатом складывания обеих критериев минимизации, поэтому, для краткости и избегания повторений формул, запишем новую целевую функцию [9]:

Новая система ограничений останется практически неизменной, лишь дополнится ограничением небаланса системы относительно центра отсчета координат :

, ,

, , ,

.

Получив формулы в общем виде не составит труда реализовать алгоритм численного решения задачи при помощи ЭВМ с помощью определенного алгоритма [10].

## Выводы к разделу 1

Проанализировав источники научной литературы, которые были отобраны относительно данной тематики, был произведен анализ предметной области поставленной задачи, было установлено набор необходимых и блок необязательных входных данных, было определено пространство решения задачи, что дает нам право использовать геометрические аксиомы указанного пространства.

По мере анализа задачи многокритериальной задачи о минимизации длины связывающей сети с учетом небаланса системы было определено необходимые и достаточные ограничения компоновочной системы. В результате построения математической модели (раздел 1), показано, что задачи компоновочного синтеза в указанной выше постановке задачи имеет широкое применение в задачах компоновки радиоэлементов на системных платах, в космической технике, в задач погрузки грузов на корабли и транспортные средства требующие баланса системы. Однако для размещения объектов круговой и прочих структур необходимо использовать специфические алгоритмы или сторонние библиотеки, которые позволят произвести оптимизационные расчеты при решении поставленной задачи на ЭВМ [11].

В пределах теории геометрического проектирования остается открытым вопрос построения математических моделей и разработки эффективных методов решения задач упаковки 3D-объектов с учетом разнообразия пространственных форм объектов и контейнеров, в которые производится упаковка, ограничений поведения, минимально и максимально допустимых расстояний между объектами и особенностей размещения объектов внутри контейнера.

# 2 ОПИСАНИЕ МЕТОДА РЕШЕНИЯ

## 2.1 Выбор и сравнение программных пакетов для решения задачи

В данной дипломной работе рассматривается задача балансной компоновки с критерием минимизации длины связующей сети в следующей постановке: разместить 3D-объекты указанного количества, которые имеют природу шара, в контейнер такой же формы с учетом специальных ограничений так, чтобы функция цели достигала своего локально-экстремального значения.

Для решения задач нелинейного программирования необходим алгоритм решения, но проблема в том, что для решения такой задачи нужно учитывать множество ограничений, критериев, ограничений переменных, а реализация такой сложной системы выльется в отдельный программный продукт, соответственно, было предложено использовать некоторый платный или бесплатный пакет для данного класса задач. В виду дороговизны коммерческих решателей, было принято решение рассмотреть класс солверов из категории «в свободном доступе». Задачей этого раздела будет поиск подходящего пакета-решателя, который бы отлично можно было импортировать на платформу программирования .Net, язык программирования C#, более детальное описание процесса выбора языка программирования будет предоставлено в разделе 3. Приступим к рассмотрению солверов.

Первый в очереди пакет Netlib, содержится по следующему электронному адресу <ftp://www.netlib.org/opt/index.html> и доступен для свободного скачивания с последующим внедрением в свой программный код; по ссылке имеются следующие алгоритмы:

1. conmax (минимизация функции с нелинейными ограничениями);
2. donlp2 (нелинейная оптимизация для дифференцируемых функций);
3. dqed (метод наименьших квадратов для нелинейных функций при линейных ограничениях);
4. hooke (оптимизация без ограничений и без использования производных);
5. lbfgs (нелинейная оптимизация при ограничениях простой структуры вида ;
6. lsnno (нелинейная оптимизация при линейных сетевых ограничениях);
7. praxis (оптимизация без ограничений и без использования производных);
8. simann (оптимизация без ограничений с использованием метода отжига Simulated Annealing);
9. subplex (оптимизация без ограничений);
10. tn (Метод Ньютона для оптимизации без ограничений или с простыми ограничениями);
11. varpro (метод наименьших квадратов для сепарабельных функций);

Разработка пакета началась в 1980 году на языках С++ и Fortran. Такой пакет собрал в себе множество алгоритмов решения, но его минусом есть оптимизационные алгоритмы в виду старины написания имеют мало функционала, некоторые не позволяют работать с ограничениями. Также, была выявлена сложность подключения на платформу .NET.

Вторым рассматриваемым пакетом стал [TRON](http://www-unix.mcs.anl.gov/~more/tron/) по ссылке <http://www-unix.mcs.anl.gov/~more/tron/>. TRON - это метод доверия Ньютона для решения больших задач ограниченной оптимизации. TRON использует метод проекции градиента для генерации шага Коши, предопределенного метода сопряженного градиента с неполной факторизацией Холецкого для генерации направления и прогнозируемого поиска для вычисления шага. Использование проецируемых поисков, в частности, позволяет TRON исследовать грани допустимого множества путем генерации небольшого количества второстепенных итераций даже для задач с большим числом переменных. В результате TRON замечательно эффективен при решении больших задач ограниченной оптимизации [12].

Этот пакет имеет ограниченное количество ограничений, что связывает нам руки при решении задач больших размерностей, еще этот пакет является «условно» бесплатным, так как, имеет ограниченный срок использования бесплатной версии. Из достоинств можно перечислить:

1. Нет предположений о строгой взаимодополняемости.
2. Глобальная конвергенция; быстрая локальная сходимость.
3. Идентификация оптимальной грани в конечном числе итераций.
4. Неполная факторизация Холецкого с предсказуемыми требованиями к хранению.

Третьим претендентом стал алгоритм L-BFGS-B метод с ограниченной памятью. представляет собой квази-ньютонный код ограниченной памяти для ограниченной оптимизации, т. е. для задач, где единственные ограничения имеют вид . Начальные реализации начинались еще в 1980 году, затем продолжились в век компьютерной автоматизации вычислений и массовой интеграции компьютеров в повседневную жизнь научно-технического прогресса.

Текущий выпуск - версия 3.0. Последний файл был изменен 02/08/11. Это программное обеспечение свободно доступно, но правообладатели ожидают, что все публикации, описывающие работу с использованием этого программного обеспечения или все коммерческих продукты, использующих его, будут цитировать, по крайней мере одну из ссылок размещенных на официальном сайте. Это программное обеспечение выпущено в соответствии с «Новой лицензией BSD» (также называемой «Измененная лицензия BSD» или «Лицензия 3-оговорки»). Данный итерационный метод численной оптимизации, предназначенный для нахождения локального максимума/минимума нелинейного функционала без учета ограничений, что не удовлетворяет описанию математической модели нашей задачи, хотя данный алгоритм очень хорош при решении очень огромных нелинейных задач, как бы мы не старались применение невозможно [13].

Рассмотрим группу следующих мощных решателей в среде разработки Matlab. Имеется множество пакетов способствующих написанию программ с использованием методов оптимизации. Например, Optimization Toolbox содержит алгоритмы для задач НЛП, Пакеты [TOMLAB](http://tomlab.biz/), [LOQO](http://www.orfe.princeton.edu/~loqo/) содержат MATLAB-интерфейс для квадратичного программирования, MCS – пакет решения задач глобальной оптимизации, работающий в среде MATLAB.

Оптимизация Toolbox предоставляет функции для поиска параметров, которые минимизируют или максимизируют цели при удовлетворении ограничений. Набор инструментов включает в себя решатели для линейного программирования (LP), смешанного целочисленного линейного программирования (MILP), квадратичного программирования (QP), нелинейного программирования (NLP), ограниченных линейных наименьших квадратов, нелинейных наименьших квадратов и нелинейных уравнений. Вы можете определить свою проблему оптимизации с функциями и матрицами или путем указания переменных выражений, которые отражают лежащую в основе математику [14].

Вы можете использовать решатели инструментария для поиска оптимальных решений для непрерывных и дискретных задач, выполнения компромиссных анализов и включения методов оптимизации в алгоритмы и приложения. Панель инструментов позволяет выполнять задачи оптимизации проекта, включая оценку параметров, выбор компонентов и настройку параметров. Его можно использовать для поиска оптимальных решений в таких приложениях, как оптимизация портфеля, распределение ресурсов, планирование производства и планирование.

TOMLAB разработан в 1999 году и продолжает расширять свой функционал, предлагает широкий спектр инструментов оптимизации, которые поддерживают глобальную оптимизацию, целочисленное программирование, все типы наименьших квадратов, линейное, квадратичное и неограниченное программирование для задач оптимизации MATLAB. Доступен для 32 и 64-разрядных платформ в Windows, Linux и MAC OS X. Попробовать решатели CPLEX, GUROBI для ваших задач MILP и MIQP можно в любой момент при наличии лицензии на среду разработки Matlab.

Список методов решения:

1. Tlsqr: Крупномасштабные разреженные линейные наименьшие квадраты.
2. glcCluster: Смешанная целая нелинейная глобальная оптимизация.
3. glcDirect: Модифицированная реализация C метода DIRECT.
4. PDCO: Метод первично-двойного барьера для выпуклых целей, управляет линейными ограничениями.
5. slsSolve: Редкие наименьшие квадраты с нелинейными ограничениями.

TOMLAB обрабатывает широкий спектр классов задач, среди которых:

1. Линейное программирование
2. Квадратичное программирование
3. Нелинейное программирование
4. Смешанное целое программирование
5. Смешанное целочисленное квадратичное программирование с или без выпуклых квадратичных ограничений
6. Смешанное целое нелинейное программирование
7. Линейные и нелинейные наименьшие квадраты с L1, L2 и бесконечной нормой
8. Экспоненциальный набор данных
9. Глобальная оптимизация
10. Полу специфическая задача программирования с билинейными матричными неравенствами
11. Ограниченное достижение цели
12. Геометрическое программирование
13. Генетическое программирование

TOMLAB - это универсальная, открытая и интегрированная среда разработки Matlab для исследований и обучения в области оптимизации на Unix и ПК. Одной из причин мотивации TOMLAB является упрощение исследований по практическим проблемам оптимизации, что дает легкий доступ ко всем типам решателей; в то же время имея полный доступ к силе Matlab. Принцип дизайна: определить вашу проблему один раз, оптимизировать с помощью любого подходящего решателя. В этой статье мы обсудим дизайн и содержание TOMLAB, а также некоторые приложения, в которых TOMLAB был успешно применен. TOMLAB основан на NLPLIB TB, наборе инструментов Matlab для нелинейного программирования и оценки параметров и OPERA TB 1.0, набор инструментов Matlab для линейной и дискретной оптимизации. Реализовано более 65 различных алгоритмов и графических утилит. Вы можете вызвать решатели в Matlab Optimization Toolbox и универсальных решателях, реализованных в Fortran или C, используя интерфейс MEX-файла [15].

LOQO - мощный решатель для задач с гладкой ограниченной оптимизацией, основанный на методе внутренней точки, применяемом к последовательности квадратичных приближений. В соответствии с требованием, чтобы определяющие функции были гладкими (в точках, оцененных алгоритмом), LOQO может обрабатывать ряд задач: линейный или нелинейный, выпуклый или невыпуклый, ограниченный или неограниченный. Для выпуклых задач LOQO находит глобально оптимальное решение; в противном случае он итерации с данной начальной точки, чтобы найти локально оптимальное решение.

Заключение:

Разработчик: проф. Роберт Вандербей, Принстонский университет

Текущая версия: 7.03

Поддерживаемые типы проблем:

1. линейные,
2. квадратичные
3. гладкие нелинейные задачи
4. ограничения непрерывных переменных.

Доступные алгоритмы: метод неразрушающего начала, первично-двойной метод внутренней точки для линейного и квадратичного программирования с надежными расширениями к более общим нелинейным объектным и ограничивающим функциям.

Особенности: Возможность утверждать, что проблема выпуклая, для большей эффективности.

LOQO - это система для решения задач с гладкой сдержанной оптимизацией. Проблемы могут быть линейными или нелинейными, выпуклыми или невыпуклыми, ограниченными или неограниченными. Единственное реальное ограничение состоит в том, что функции, определяющие задачу, являются гладкими (в точках, оцененных алгоритмом). Если проблема выпукла, LOQO находит глобально оптимальное решение. В противном случае он находит локально оптимальное решение рядом с заданной начальной точкой.

MCS - это программа Matlab для связанной ограниченной глобальной оптимизации, использующая только значения функций, основанные на многоуровневом поиске координат, который уравновешивает глобальный и локальный поиск. Локальный поиск выполняется посредством последовательного квадратичного программирования.

Многоуровневый поиск координат (MCS) - это алгоритм для связанной ограниченной глобальной оптимизации с использованием только значений функций. Для этого n-мерное пространство поиска представлено набором непересекающихся гиперкубов (ящиков). Затем ящики итеративно расщепляются вдоль плоскости оси в соответствии со значением функции в представительной точке ящика и размером коробки. Эти два критерия разделения объединяются, чтобы сформировать глобальный поиск, разбивая большие поля и локальный поиск, разделяя области, для которых значение функции является хорошим. Кроме того, для повышения производительности алгоритма можно использовать локальный поиск, объединяющий квадратичный интерполятор функции и поиск строк.

Библиотека написана на Fortran и проблема с простыми оценками с использованием многоуровневого поиска координат. Этот решатель дополняется рядом поддержки для инициализации данных и установки необязательных параметров. Производные инструменты не требуются, и

решатель предназначен для среднесрочных задач. Такой решатель будет полезен в любой из областей, в которых существующая локальная оптимизация функции, а также в тех областях, где жизненно важно найти глобальные, а не локальные. Примеры областей включают: финансовые, химические и фазово-равновесные проблемы, графический и сетевой анализ, планирование, свертывание белков и робототехника.

Выше рассмотренные методы, библиотеки, чрезвычайно эффективны и хороши в плане реализации алгоритмов, имеют ряд преимуществ на ряду с некоторыми коммерческими продуктами и в тоже время имеют один отличительный недостаток, их применение возможно в среде Matlab и только, нас же интересует .NET платформенная импортируемая алгоритмическая программная оболочка имеющая необходимый функционал для решения поставленной модельной задачи равновесной упаковки с учетом длины связующей сети.

Опишем информационные сведения относительно IPOPT (<http://www-124.ibm.com/developerworks/opensource/coin/Ipopt>) – использует алгоритм внутренней точки для задач НЛП большой размерности. Свободно распространяется, имеет открытый исходный код и не требует обязательств при создании коммерческих программ. Первоначальная версия этого документа была создана в качестве учебного проекта, преподаваемого профессором Франсуа Марго в Университете Карнеги-Меллона. После этого добавлено значительные участки кода.

Оригинальная версия IPOPT (версия Fortran) была продуктом диссертационного исследования Андреаса Вахтера под руководством Лоренца Т. Биглера на факультете химической инженерии Университета Карнеги-Меллона. Код был создан с открытым исходным кодом и распространен по инициативе COIN-OR, которая теперь является некоммерческой корпорацией. IPOPT активно развивается под COIN-OR с 2002 года.

Чтобы продолжить естественное расширение кода и позволить легко добавлять новые функции, IBM Research решила инвестировать в переиздание с открытым исходным кодом IPOPT на C ++. С помощью Карла Лайда, который пришел в отдел математических наук в IBM Research в качестве летнего стажера в 2004 и 2005 годах во время его исследований в PhD, код был повторно внедрен с нуля.

Новая версия кода оптимизации IPOPT (IPOPT 3.0.0 и выше) была сохранена в IBM Research и остается частью инициативы COIN-OR. Разработка версии Fortran прекратилась, но исходный код можно загрузить с <https://www.coin-or.org/Ipopt/ipopt-fortran.html>. Данный метод пакета-библиотеки позволяет применять для решения оптимизационных задач размещения методы оптимизации IPOPT отлично импортируется на все языки платформы .NET, так как, реализован на С++, соответственно применим с использованием C#. Данный солвер имеет множество настроек работы алгоритма, включает множество встроенных методов, которые не нуждаются в повторной реализации и отлично оптимизированы по скорости выполнения, простота освоения его сильная сторона. Документация написана в полном объеме и доступна на официальном сайте без ограничений доступа.

Проведя качественный анализ всех бесплатных пакетов для решения задач нелинейного оптимизации, после взвешиваний всех особенностей и сильных сторон каждого из решателей было выбрано IPOPT с алгоритмом внутренней точки как основной метод получения решения при решении задачи дипломного проекта.

## 2.2 Описание метода поиска локального минимума

Как было сказано в пункте 2.1 раздела 2 при рассмотрении существующих пакетов оптимизации с открытым исходным кодом и публичной лицензией, мы остановили свой выбор на IPOPT. Данный пакет реализует метод внутренней точки. Перейдем к рассмотрению особенностей и преимуществ данного метода.

Начиная с выдающейся работы Н. Кармаркара, опубликованной в 1985г., и примерно до 2000г. развитие теории и методов оптимизации было в основном связано с прогрессом в теории полиномиальных методов внутренней точки. Были получены новые и очень эффективные методы решения задач линейного программирования, которые существенно подняли планку соревнования с традиционным симплекс-методом. Была разработана общая теория самосогласованных функций, которая позволяла строить полиномиальные методы внутренней точки для всех выпуклых задач с явной структурой. Появилась теория конических нелинейных задач, в рамках которой удалось построить длинно шаговые прямо-двойственные методы внутренней точки, способные ускоряться и стартовать из недопустимых точек.

Новые эффективные методы появились в квадратичном и полуопределенном программировании. Очень важно, что прогресс в этом направлении сопровождался и инициировался детальным теоретическим анализом эффективности предлагаемых алгоритмов. Оценки скорости сходимости стали важным аргументом в пользу новых методов. Может быть даже более важным, чем результаты предварительных численных экспериментов. К концу этого периода выяснилось, что все это время запросы пользователей росли быстрее, чем возможности новых методов.

Прогресс в развитии скорости вычислений компьютеров, увеличение объемов оперативной памяти, возможность использовать вспомогательные программ, небывалая доступность различной информации через Интернет, дешевизна и доступность вычислительных средств – все это вместе на порядки упростило процесс создания математических моделей и, как неизбежное следствие, существенно увеличило их размеры. Худшей ситуации для методов внутренней точки придумать было нельзя, так как трудоемкость этих методов растет (по крайней мере) пропорционально кубу размерности [16].

Среди современных методов решения задач оптимизационного программирования, методы внутренней точки (МВТ) рассматриваются как наиболее эффективные. Однако эти методы требуют очень много вспомогательных вычислений. Для задач размера они находят решение задачи с указанной точностью за:

Понятно, что такая трудоемкость итерации дает большие шансы на успех градиентным методам, у которых каждая итерация намного дешевле. Однако основным недостатком градиентных методов является их медленная сходимость. Субградиентный метод для общих выпуклых задач требует итераций для нахождения решения с точностью. В этой оценке сильная зависимость от точности усугубляется присутствием некоторой константы , которая зависит от нормы матрицы ограничений, размера решения, качества стартовой точки, и т.д. Эта константа может оказаться неожиданно большой. Поэтому в настоящее время градиентные методы составляют конкуренцию МВТ только на очень больших задачах [17].

Однако было доказано, что правильное использование структуры задач может дать алгоритмы градиентного типа дает итераций.

Более того, было доказано что для некоторых задач константу можно найти в явном виде и что она достаточно мала. Этот результат был распространен на методе решения некоторых задач минимизации с относительной точностью. А именно, было показано, что решение некоторых задач с относительной точностью δ может быть найдено за итераций градиентного типа. Заметим, что во многих приложениях концепция относительной точности очень привлекательна, так как точность решения автоматически подстраивается к любому размеру оптимального решения.

Таким образом, в этом случае не надо опасаться больших неизвестных констант. Более того, во многих случаях большая относительная точность просто не нужна - порядок в 0.5%− 0.1% вполне достаточен. Однако такой подход применим к специальным коническим задачам безусловной минимизации. Это задачи минимизации выпуклой неотрицательной однородной функции , на подпространстве, не содержащем нуля. Для решения этой задачи с относительной точностью нужно знать пару подобных концентричных аппроксимирующих эллипсоидов (эллипсоидов Джона) для субдифференциала целевой функции в нуле.

М ы рассматривали два нетривиальных пакета Netlib и TRON, которые позволяют решать поставленные задачи с относительной точностью δ за итераций градиентного типа [18].

Заметим, что объем предварительных вычислений для обоих методов (градиентного и МВТ) вполне приемлем: он находится на уровне операций в худшем случае. С точностью до логарифмического фактора, эта оценка совпадает с трудоемкостью нахождения проекции точки в на линейное подпространство, с заданной системой из уравнений.

Последующий процесс минимизации, как правило, оказывается более дешевым. Действительно, одна итерация градиентного типа требует операций. Поэтому, всему методу нужно порядка операций, но это все в теории, а на практике оказывается больше в раза 2-3 итеративных проходов при плотно заполненных матрицах ограничений, что делает градиентные методы более уязвимыми по сравнению с методами внутренней точки.

Рассмотрим методы внутренней точки. Рассматривается выпуклая задача условной оптимизации:

|  |  |
| --- | --- |
|  | (1) |

Здесь матрица А имеет ранг p, все функции являются выпуклыми и непрерывно-дифференцируемы. Обозначим через – доменное множество, через – множество допустимых решений, а через – решение задачи (1). Заметим, что в выпуклых задачах условной оптимизации ограничения вида равенства могут быть только линейными, иначе допустимое множество не будет выпуклым.

Рассмотрим функцию Лагранжа:

Если точка является (глобальным) решением задачи (1), то найдутся такие, что удовлетворяют системе Куна-Таккера:

|  |  |
| --- | --- |
|  | (2) |

Рассмотрим двойственную задачу функции Лагранжа и двойственную задач оптимизации:

|  |  |
| --- | --- |
|  | (3) |

Здесь и далее будем предполагать, что выполнены условия Слейтера, т.е. . В этом случае условия (2) являются достаточными условиями оптимальности для задачи (1). Кроме того, выполняется свойство сильной двойственности, т.е. решение прямой задачи совпадает с решением двойственной ). Этот факт позволяет получить неэвристический критерий останова для метода решения задачи (1), который итерационно вычисляет как прямую точку x, так и двойственную точку такую, что . Тогда величина будет верхней оценкой на , и итерационный процесс оптимизации можно завершать, если удовлетворено условие [19].

Метод Ньютона для систем с ограничениями вида равенства. Рассмотрим задачу оптимизации:

|  |  |
| --- | --- |
| , | (4) |

где – выпуклая, дважды непрерывно-дифференцируемая функция.

Здесь по-прежнему присутствуют только ограничения вида равенства, поэтому выполняется свойство сильной двойственности, а условия Куна-

Таккера являются необходимыми и достаточными условиями оптимальности. Рассмотрим несколько способов решения задачи (4) [20].

Исключение неизвестных. Общее решение СЛАУ можно представить как где – частное решение СЛАУ, – матрица, в столбцах которой стоят базисные вектора для – произвольный вектор. В результате задачу (4) можно эквивалентно переписать как задачу выпуклой безусловной оптимизации:

,

Для этой задачи может быть задействован обширный арсенал методов решения задач безусловной оптимизации. Однако, такой подход может искажать структуру исходной задачи. Если, например, матрица A является сильно разреженной, то матрица G получится произвольной, и свойство разреженности не удастся использовать для более эффективного решения задачи (4).

## 2.3 Описание метода поиска глобального минимума

Рассматривается задача на примере равновесной упаковки семейства кругов в круге минимального радиуса в виде многоэкстремальной задачи нелинейного программирования. С помощью негладких штрафов задача сводится к задаче безусловной минимизации негладкой функции. Предлагается алгоритм поиска локальных экстремумов негладкой функции и алгоритм уточнения оценки снизу для значения глобального минимума целевой функции, которые базируются на использовании методов оптимизации негладких функций с применением модификации r-алгоритма Шора.

Задача равновесной упаковки неодинаковых кругов в круг наименьшего радиуса возникает в задачах плотной упаковки параллельных одинаковых по высоте круговых цилиндров в цилиндрический контейнер при ограничениях на динамическое равновесие системы. Динамическое равновесие определяется требованием, чтобы центр тяжести системы круговых цилиндров находился в центре кругового контейнера.

Математическая модель задачи равновесной упаковки неравных кругов может быть сформулирована в виде различных многоэкстремальных задач математического программирования. Одна из этих формулировок является предметом исследования данной работы. Опишем алгоритм нахождения локальных экстремумов и алгоритм уточнения оценки снизу для значения глобального минимума целевой функции, которые базируются на использовании методов оптимизации негладких функций.

Имеется семейство кругов с радиусами и весами , . Полагаем, что центр тяжести круга находится в его центре. Равновесной упаковкой семейства кругов , , в круг назовем такую их упаковку, чтобы радиус круга был минимальным и центр тяжести семейства кругов , , совпадал с центром круга S [22].

Не ограничивая общности будем считать, что центр круга находится в начале неподвижной системы координат. Пуст точка являтся – неизвестным центром круга ; – неизвестный радиус круга . Обозначим известные величины нижнюю границу на искомый радиус . Тогда равновесной упаковке семейства кругов соответствует многоэкстремальная задача нелинейного программирования [23]:

|  |  |
| --- | --- |
|  | (6) |

при следующих ограничениях:

|  |  |
| --- | --- |
|  | (7)  (8)  (9) |

где . Здесь целевая функция (6) является линейной. Ограничения (7) гарантируют, что , а ограничения (8) описывают условие , где int(·) означает внутренность множества (·).

T

Ограничения (9) означают, что центр тяжести семейства кругов , , находится в центре круга . Ограничение (7) обеспечивает то, что значение радиуса круга S не уходит к минус бесконечности, чему формально не препятствует правая часть ограничения (8) [24].

В работе приведены еще две формулировки этой задачи. Первая является задачей обратно-выпуклого программирования, а вторая — задачей минимизации функции максимума из выпуклых функций при ограничениях (8) и (9). Во второй формулировке переменная r не используется и ее оптимальное значение определяется из минимального значения негладкой целевой функции. Обе формулировки свободны от ограничения, так как не отрицательность r учитывается за счет формулировки ограничения (6) в виде:

|  |  |
| --- | --- |
|  | (10) |

Алгоритм поиска наилучшего решения включает в себя сведение задачи (6)–(9) к задаче безусловной оптимизации негладкой функции[2:

|  |  |
| --- | --- |
|  | (11) |

где штрафная функция имеет вид:

|  |  |
| --- | --- |
|  | (12) |

Здесь положительные штрафные коэффициенты, а функции определяются так:

|  |  |
| --- | --- |
|  | (13) |

где – заданные допуски на отклонение координат центра тяжести семейства кругов от начала координат. Использование в (7) штрафных коэффициентов позволяет учесть точность выполнения ограничений (6)–(7). Коэффициент согласно (8), отвечает за ограничения (2), (3), коэффициент согласно (9), – за ограничения (7), а коэффициент – за ограничение (11).

Алгоритм поиска наилучшего решения задачи (1)–(5) состоит в следующем. Для заданного набора стартовых точек осуществляется поиск локальных минимумов в задаче (6) с помощью модификации r-алгоритма. Наилучший из локальных минимумов функции , для которого штрафная функция близка к нулю, принимается за решение задачи (6)–(9). Ему соответствует значение целевой функции – наилучшее значение радиуса круга [25].

Стартовые точки генерируются случайным образом в круге заданного радиуса, который последовательно уточняется по мере нахождения лучшего локального минимума. Отметим, что данный алгоритм можно использовать и в случае, когда не требуется учитывать ограничения на центр тяжести. Для этого достаточно положить равным нулю штрафной коэффициент . Программная реализация алгоритма выполнена на некоммерческом языке GNU Octave.

Программа либо находит один из локальных минимумов в задаче (1)–(6), либо сообщает о невозможности найти допустимую точку для системы ограничений (6)–(8). Ядром программы является octave-функция ralgb5, которая реализует r-алгоритм с постоянной величиной коэффициента растяжения пространства и адаптивной регулировкой шага в направлении нормированного антисубградиента.

Регулировка направлена на увеличение точности поиска минимума функции по направлению в процессе счета и при этом гарантирует, что среднее (по итерациям) число шагов не превышает двух–трех.

## Выводы к разделу 2

В текущем разделе было рассмотрено ряд оптимизационных пакетов предоставляющих решение различных задач. Был произведен сравнительный анализ функционала пакетов в разделе 2.1, было рассмотрено достоинства и недостатки этих пакетов. В качестве применяемого в этом дипломном проекте был выбран пакет IPOPT, с алгоритмом внутренней точки, который позволяет решать оптимизационные задачи особо крупных размерностей с отличной скоростью сходимости. Был рассмотрен алгоритм поиска локального и глобально решения.

# 3 ОПИСАНИЕ ПРОГРАММНОГО ПРОДУКТА

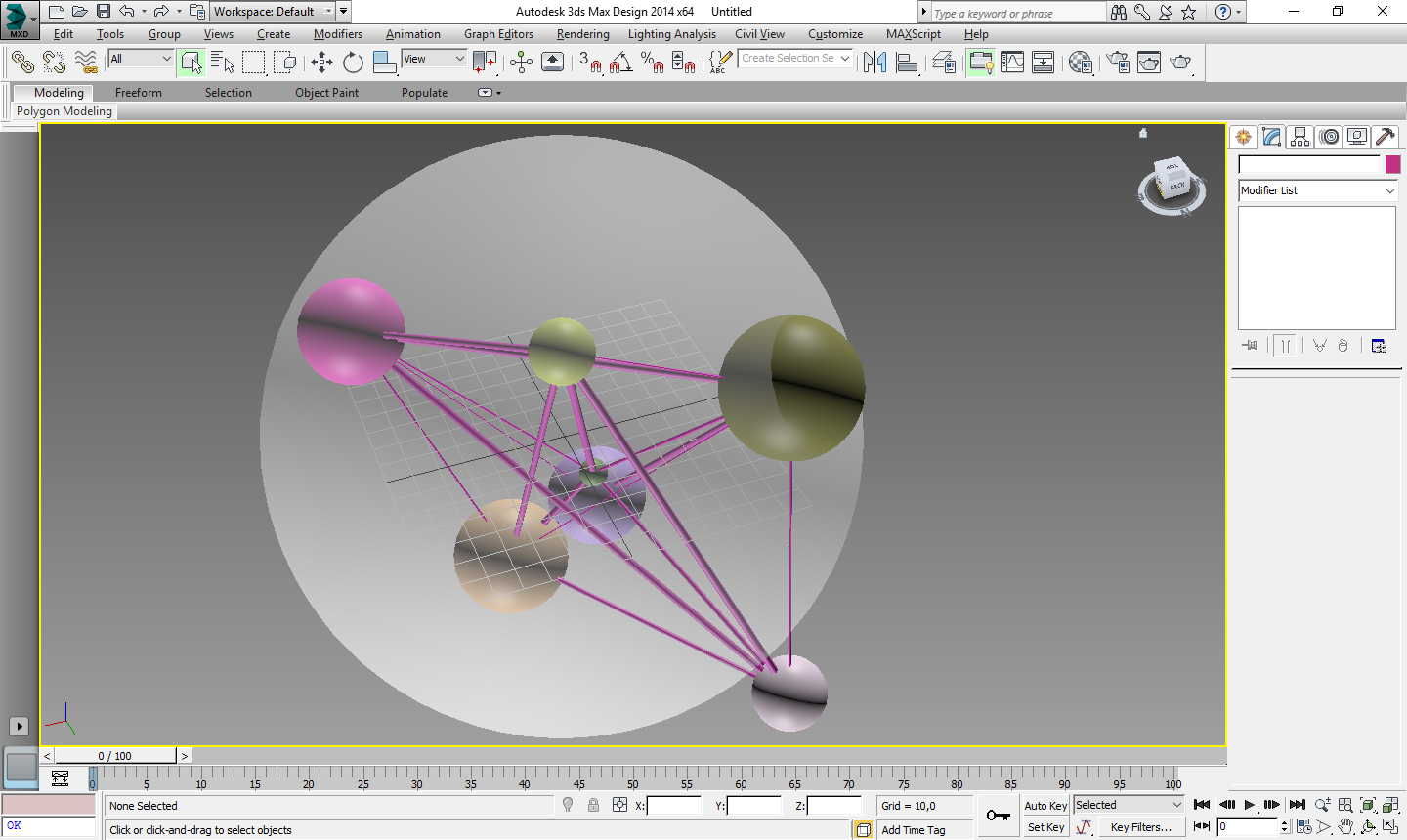
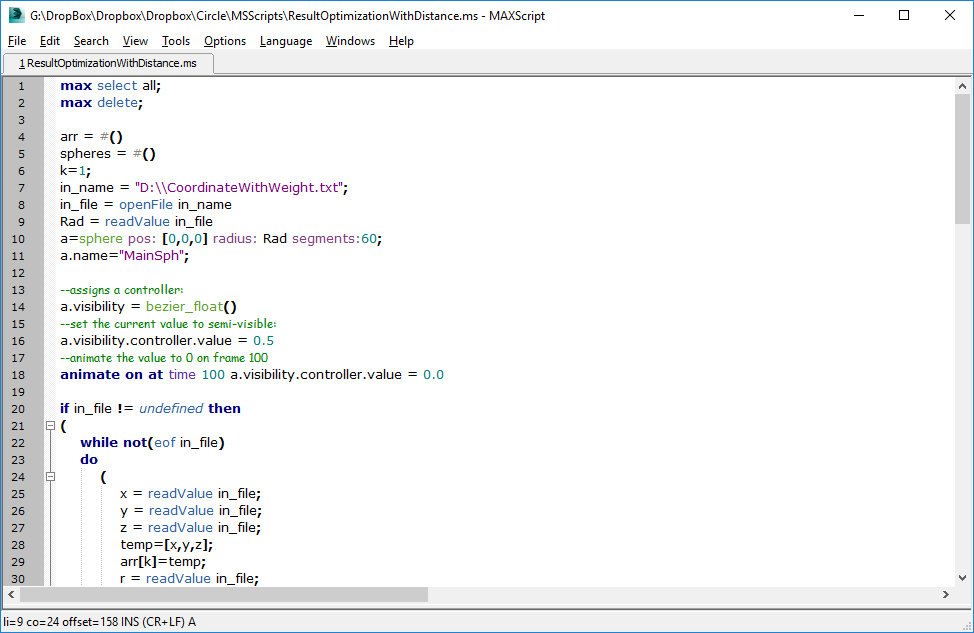


Рисунок 3.31- Начальное размещение шаров



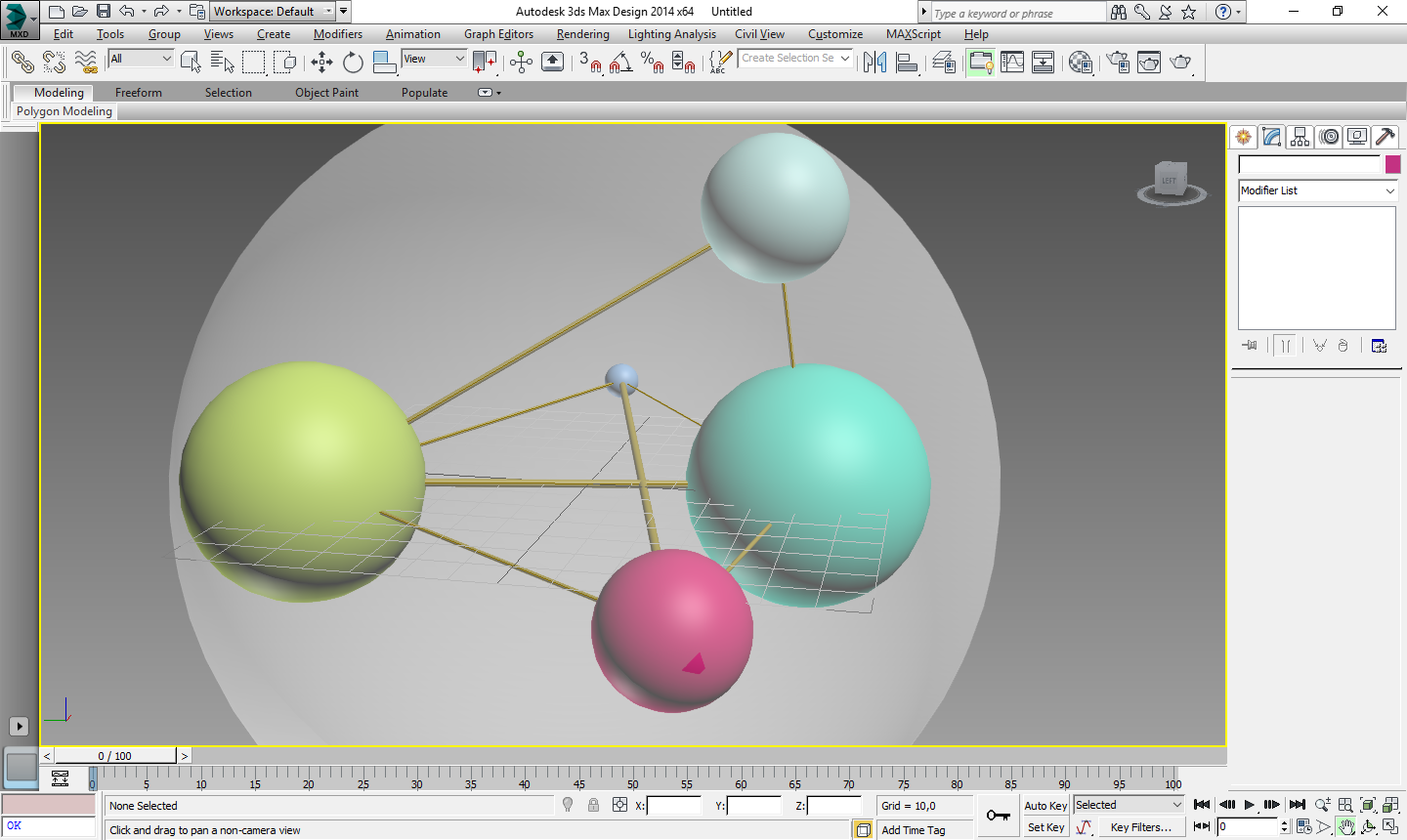


Рисунок 3.33 – Полученное решение

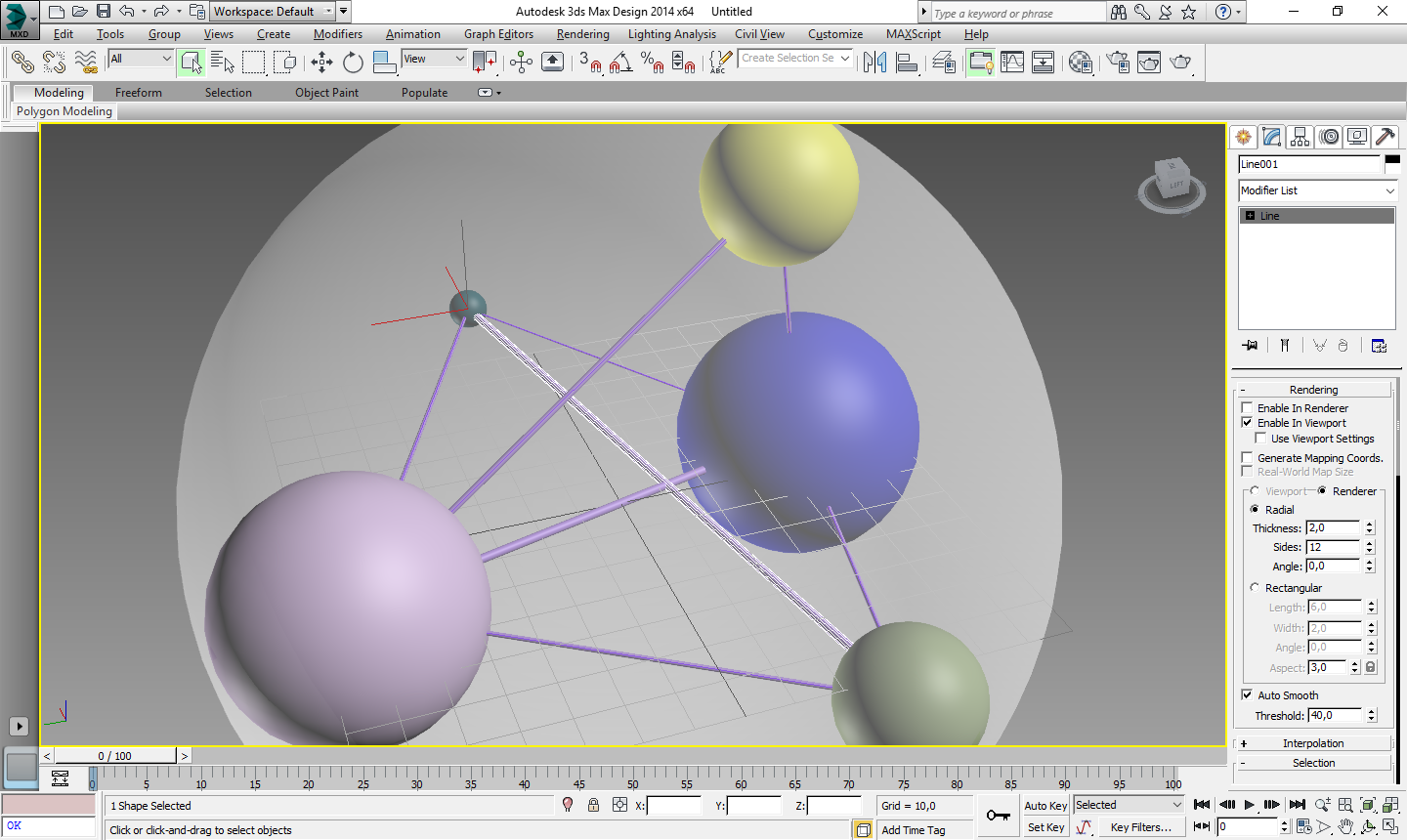


Рисунок 3.34 – Настройка свойств объекта

:

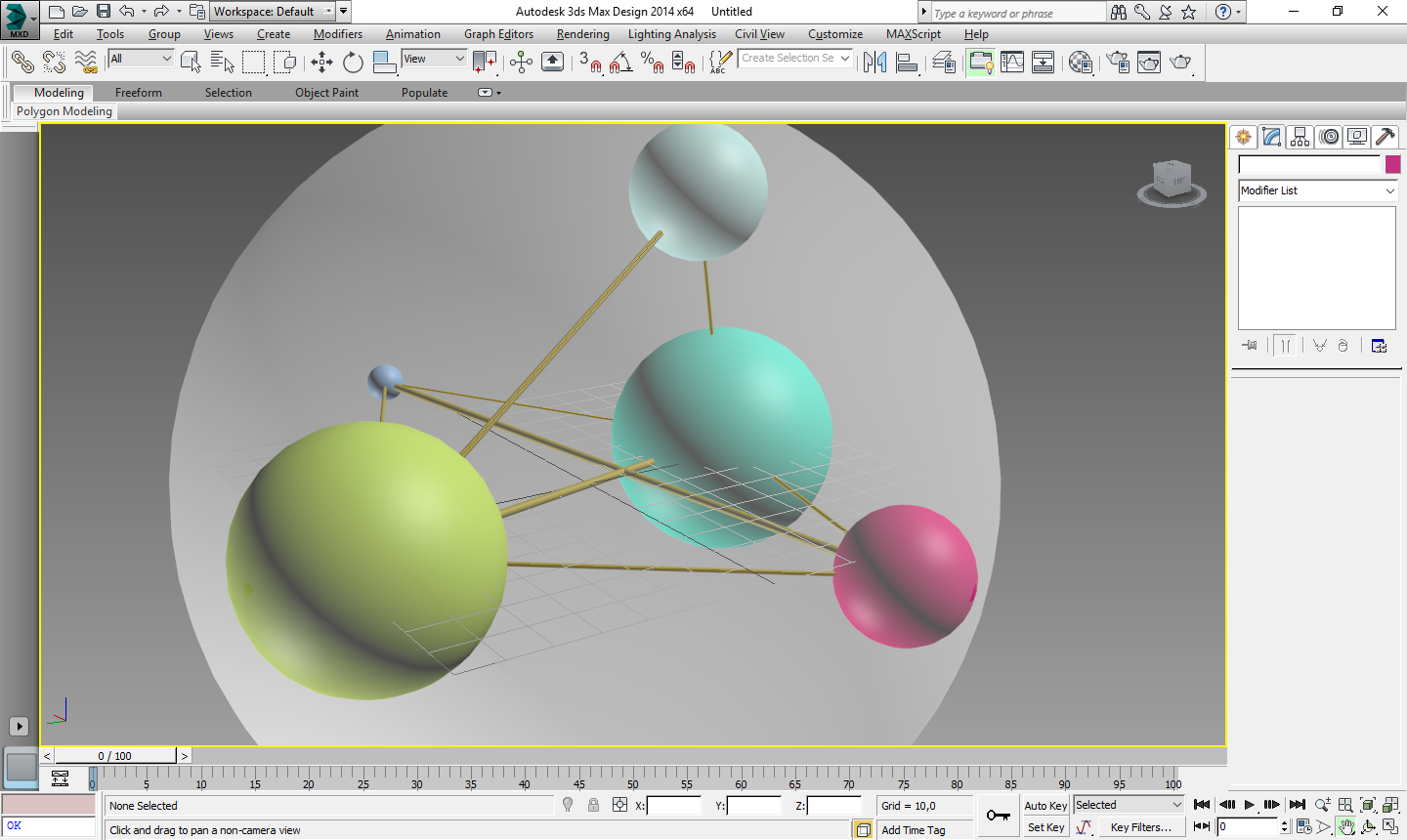


Рисунок 3.35 – Отображение полученного решения равновесной

упаковки

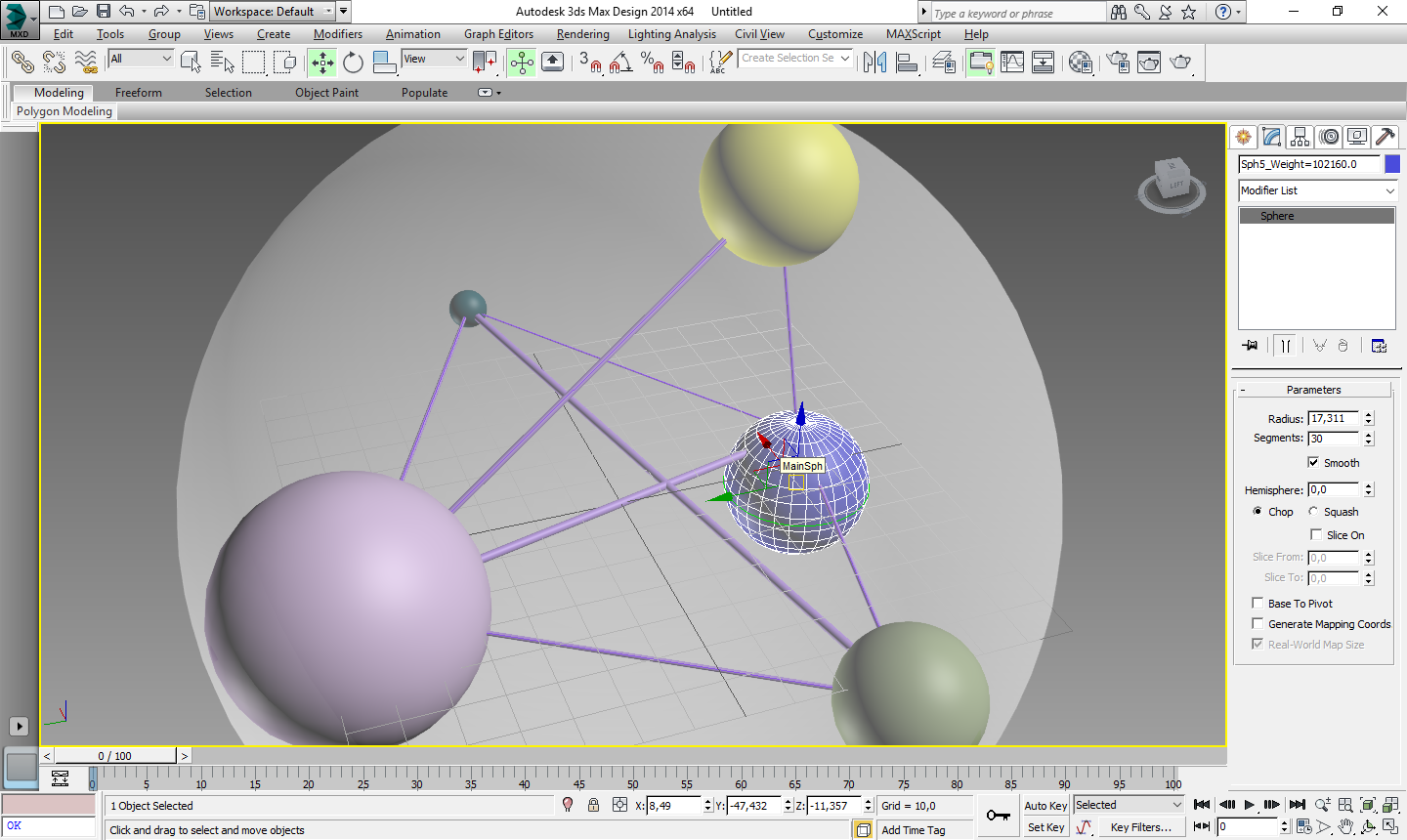


Рисунок 3.38 – Новое положение синего шара

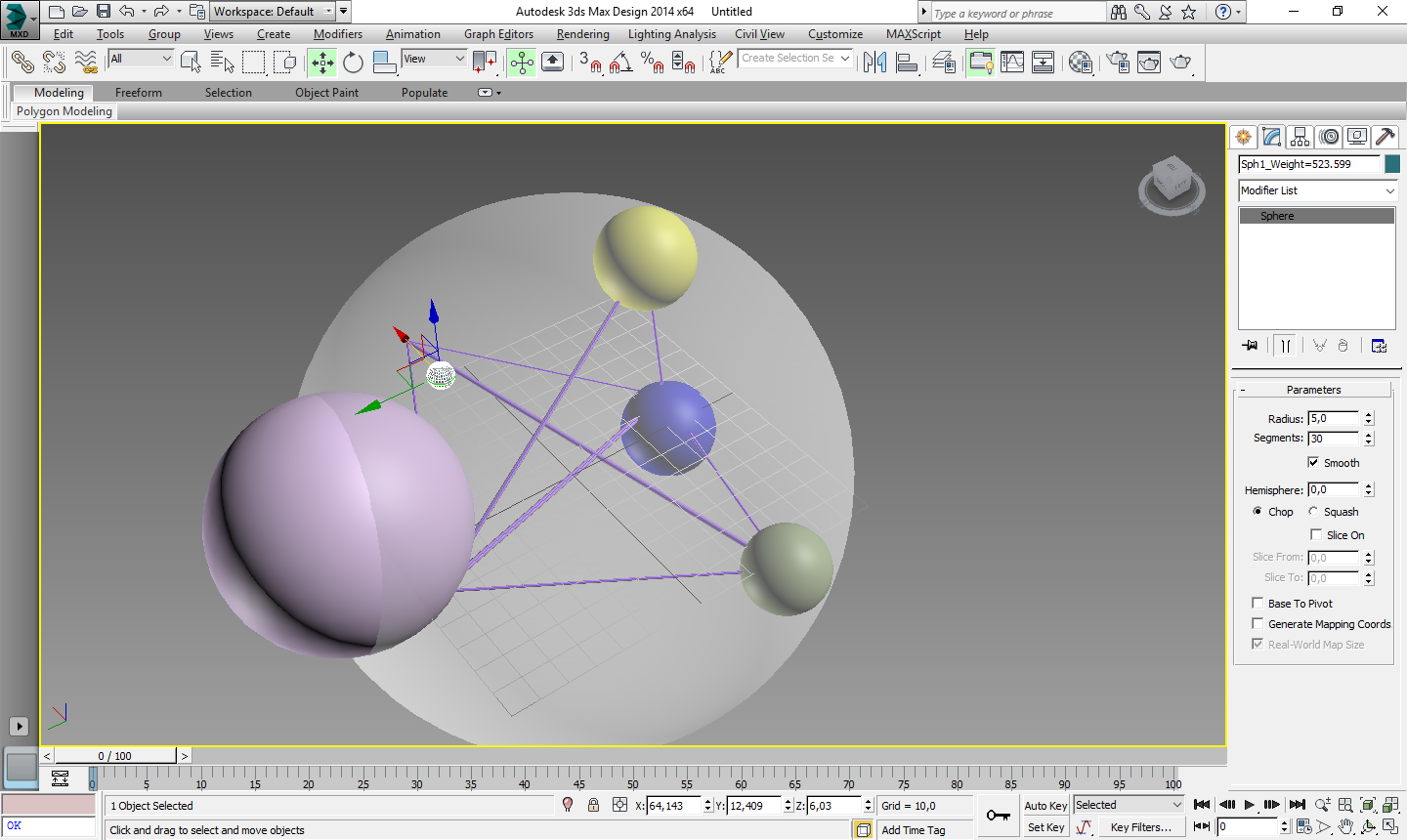


Рисунок 3.39 – Изменение пространственных характеристик нескольких шаров

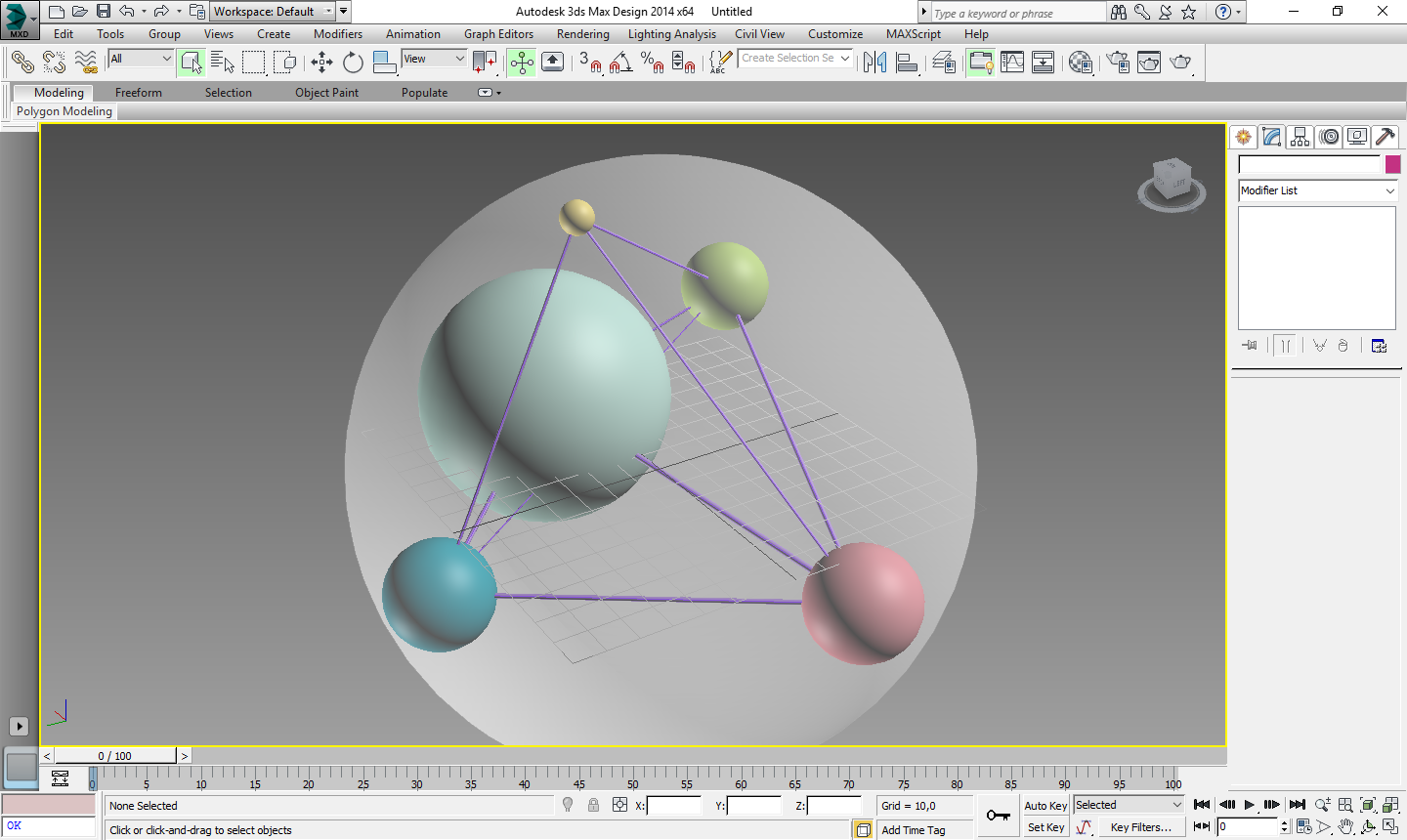


Рисунок 3.45 – Отображение в пространстве полученного нового решения

## 3.5 Результаты тестирования зависимости скорости от количества объектов

Рассматривается задача размещения различного количества шаров на каждой под итерации для каждого случая, проведено испытаний для каждого набора данных – 3-5, считалось среднее значение, после чего на каждой следующей итерации увеличивалось количество шаров на 3 единицы. (рис. 3.46):

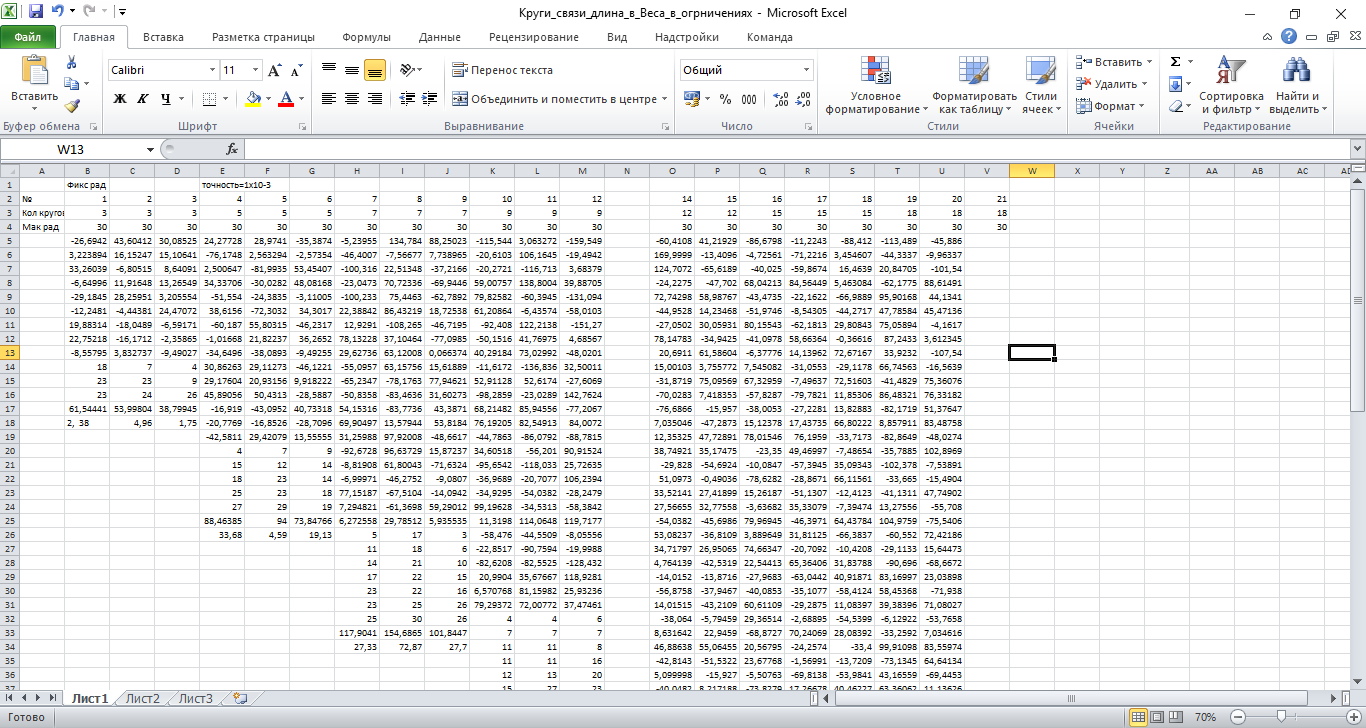


Рисунок 3.46 – Представление собранных данных

Закроем все приложения, оставив только системные. Будем замерять время использования CPU, и уровень нагрузки на CPU в процессе решения задачи.

В эксперимент используется метод фиксированного радиуса, задавать будем их при помощи генератора псевдослучайных чисел с максимальным в 30 единиц.

Рассматриваем случай с фиксированными радиусами. Будем брать от 1 до 30, в отдельном подходе будем брать до 18 кругов радиусом от 1 до 30, проводим 3 замера времени для получения среднего значения, точность решения 10е-2.

На рисунке 3.46 видим часть полученных наборов данных, где в верхней части видим детали настроек программы, внизу время получения решения, справа формат получения данных на выход пользователю для дальнейшей визуализации и отображения решения, еще ниже, расположены полученные при проведении эксперимента средние значения по времени (рис. 3.47):

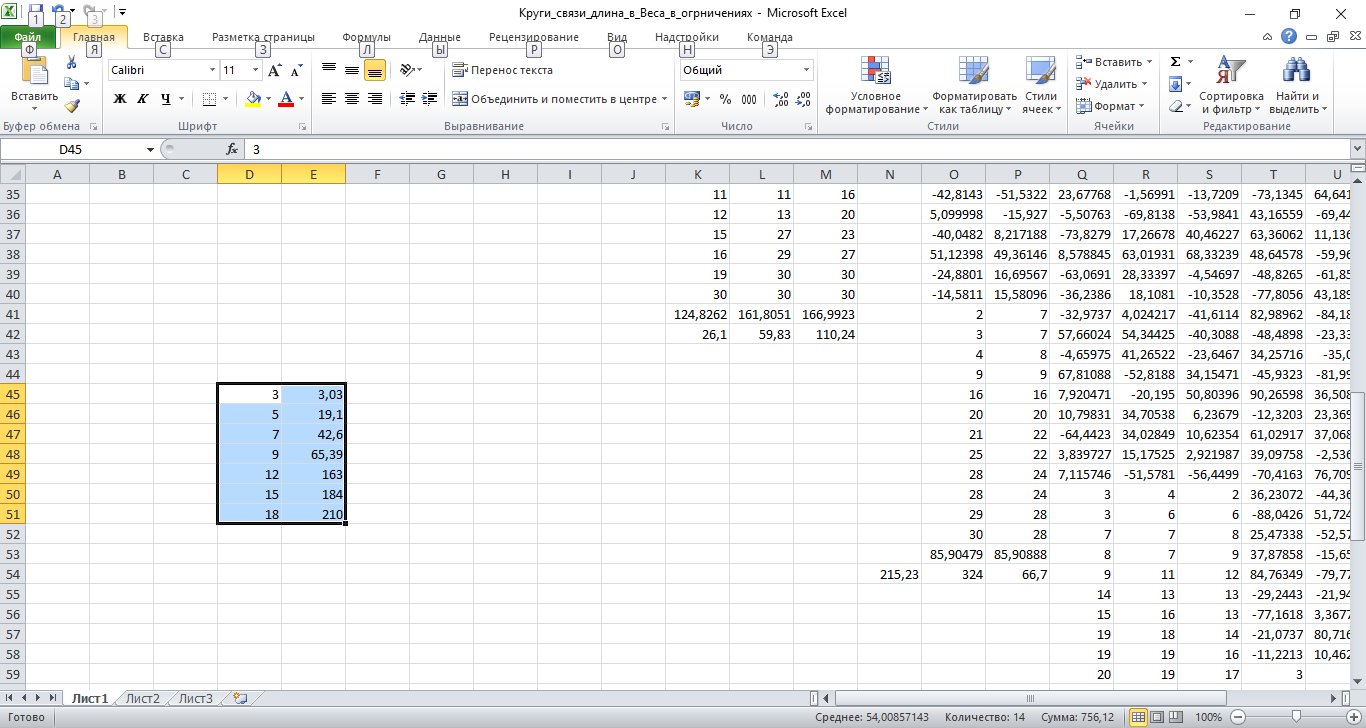


Рисунок 3.47 – Полученные средние значения временных характеристик

Загруженность центрального вычислительного компонента приведена ниже на рисунке 3.48:

Рисунок 3.3 – Нагрузка на центральный процессор при указанном числе шаров (снизу)

В результате всех экспериментов удалось проследить следующую зависимость роста времени счета в зависимости от роста количества шаров (рис. 3.49):

Рисунок 3.49 – Временная зависимость от количества шаров

График отображает некоторую кривую линию. Можно задать вполне закономерный вопрос: как программа будет себя вести в условиях необходимости решения более объемных задач, к примеру, на 50-60 шарах с учетом состояния баланса и минимальной длины связи? Странно, но иногда задачи такой размерности, при удачных наборах данных решаемы за 8-10 минут, все благодаря быстро сходящемуся методу внутренней точки. Чтобы ответить на поставленный вопрос, был сделано несколько таких замеров, результаты которых будут показаны ниже на рисунках, а сейчас посмотрим на распределение нагрузки на каждое ядро центрального процессора на диаграмме 3.50:

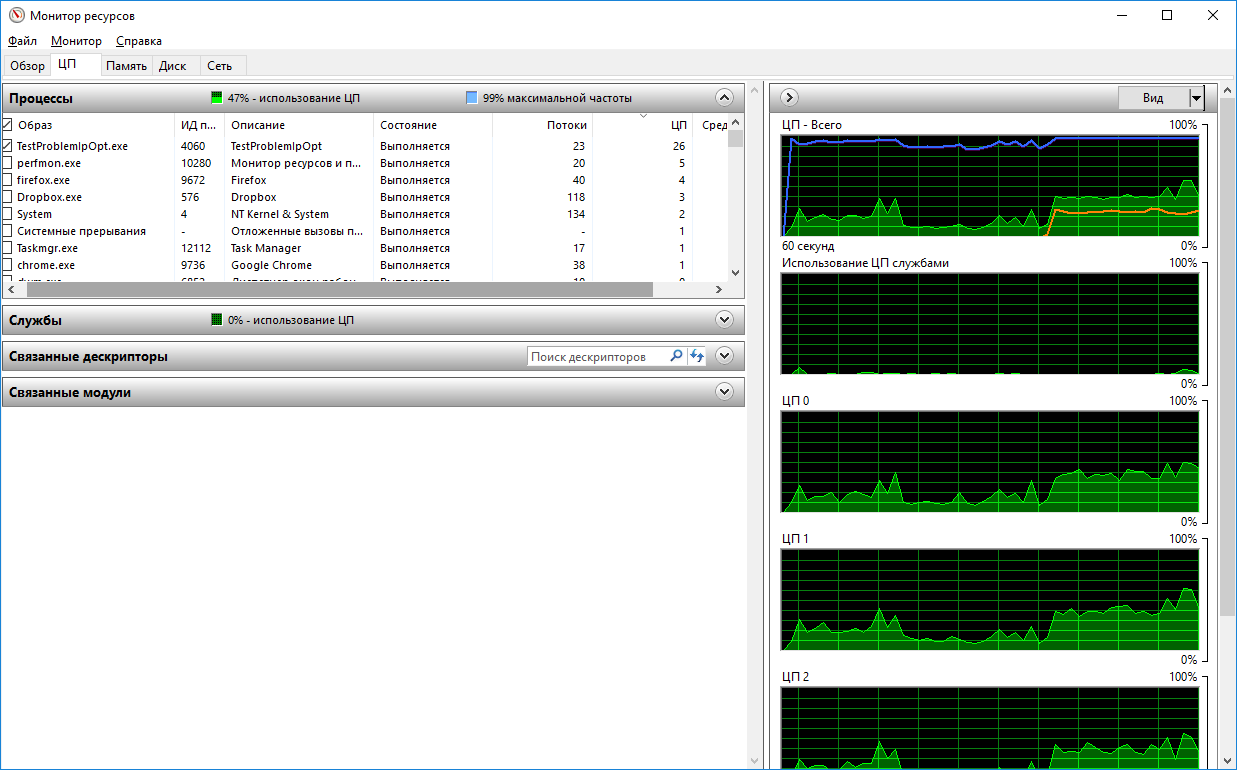


Рисунок 3.50 – распределение загрузки ядер при 5 шарах

При помощи применения параллельных циклов удалось уменьшить время выполнения поставленных задач иногда до 50% по сравнению с первоначальным временем счета (на одном ядре). Таким образом нам удалось преобразовать приложение в многопоточное ПО, которое использует распределенную нагрузку на ядра CPU. Через монитор ресурсов видны количество используемых потоков и статус завершения выполнения задачи (рис. 3.51):

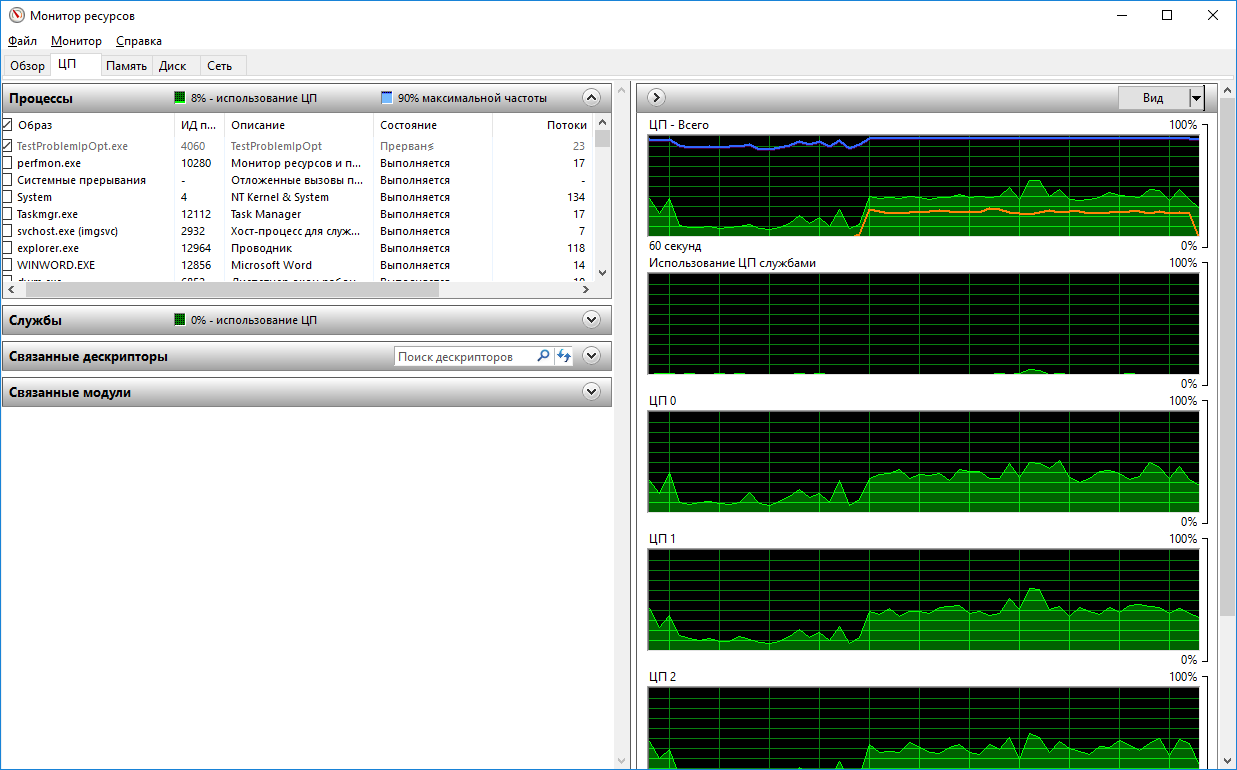


Рисунок 3.51 – Монитор ресурсов

Ниже приведено информационная таблица с графиками, на которых виднеется равномерная загрузка CPU (рис. 3.52):

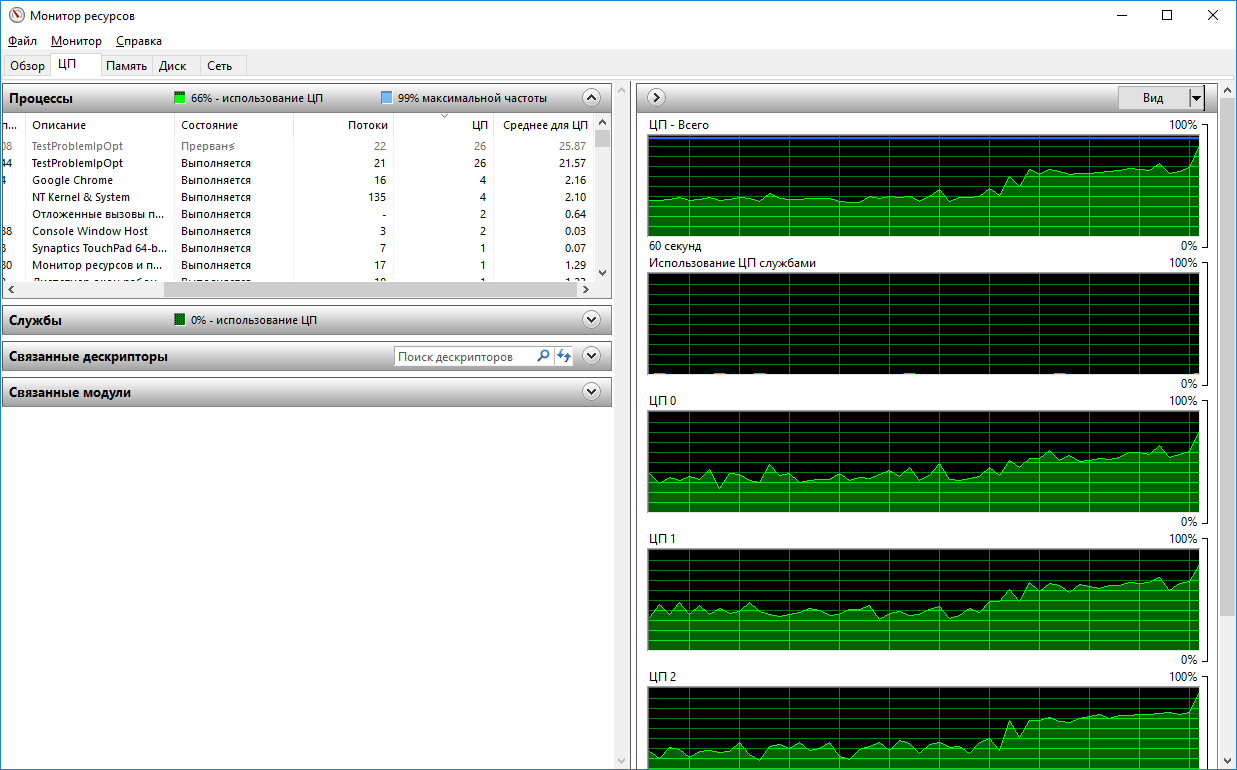


Рисунок 3.52 – Равномерная загрузка CPU

Продиагностируем встроенными средствами VS17 данный проект на расточительность памяти (рис. 3.53):

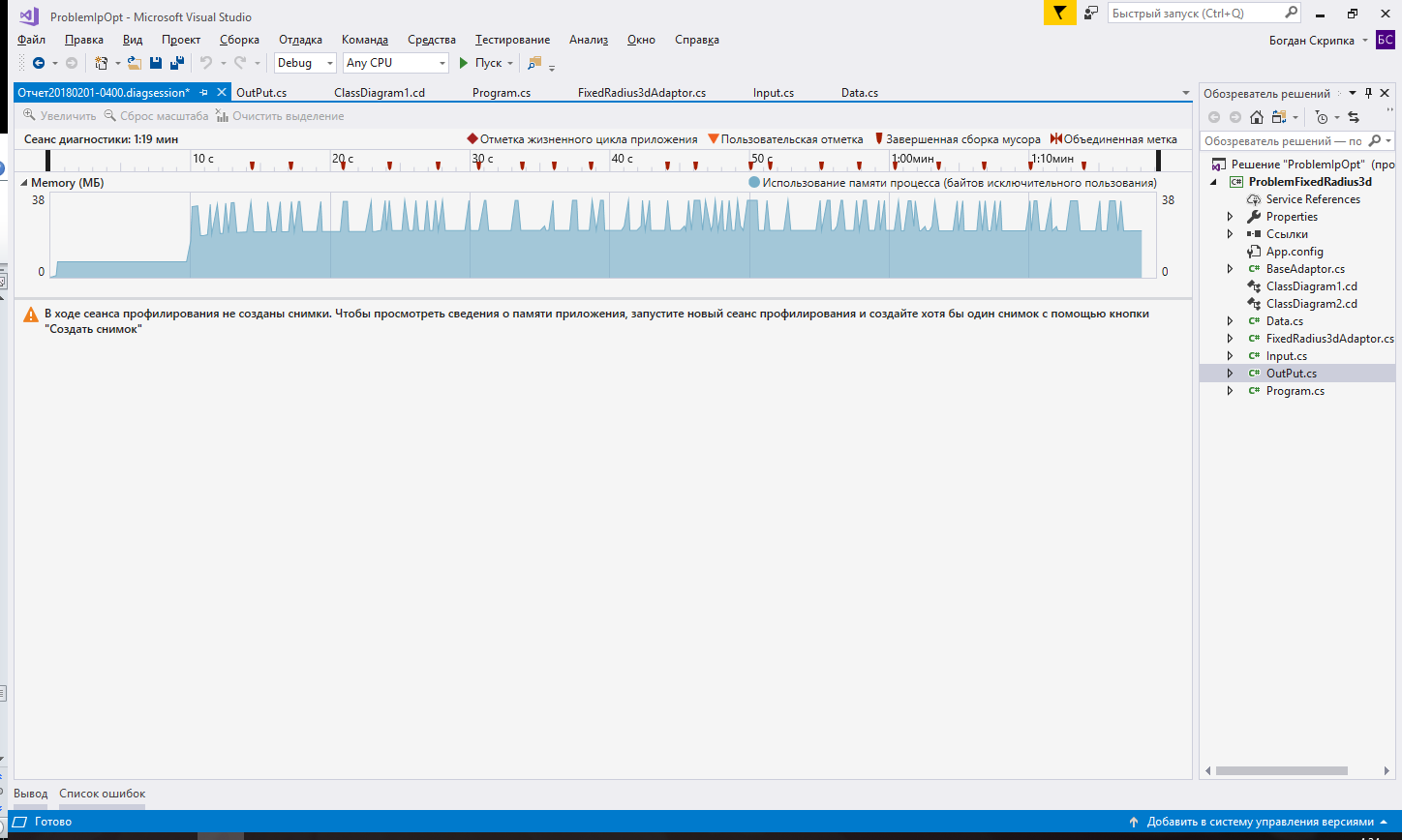


Рисунок 3.53 – График освобождения и выделения памяти

Левый плоский участок – это начало работы приложения в режиме ожидания ввода данных, затем создается экземпляр класса Ipopt и начинается решение задачи. Подымающиеся острые шпили это начало новой итерации, вычисление некоторого значения внутренней точки, сбор мусора коллектором и переход на новую итеративную ступень до выполнения условия выхода из тела программы. В пиковые моменты для задачи на 7-15 шарах требуется 33-35Мб оперативной памяти, в моменты спада это 23-25 Мб.

Полученные результаты отображают ожидания, если предварительно проанализировать данные с графиков выше. Таким образом, можно подвести краткое заключение данного эксперимента. Заключением будет следующее: было проведено множество испытаний, которые показали характер некоторой кривой зависимости для шаров с фиксированным радиусом.

А сейчас покажем качество решения на рисунках, которые были получены в процессе решения основной задачи.

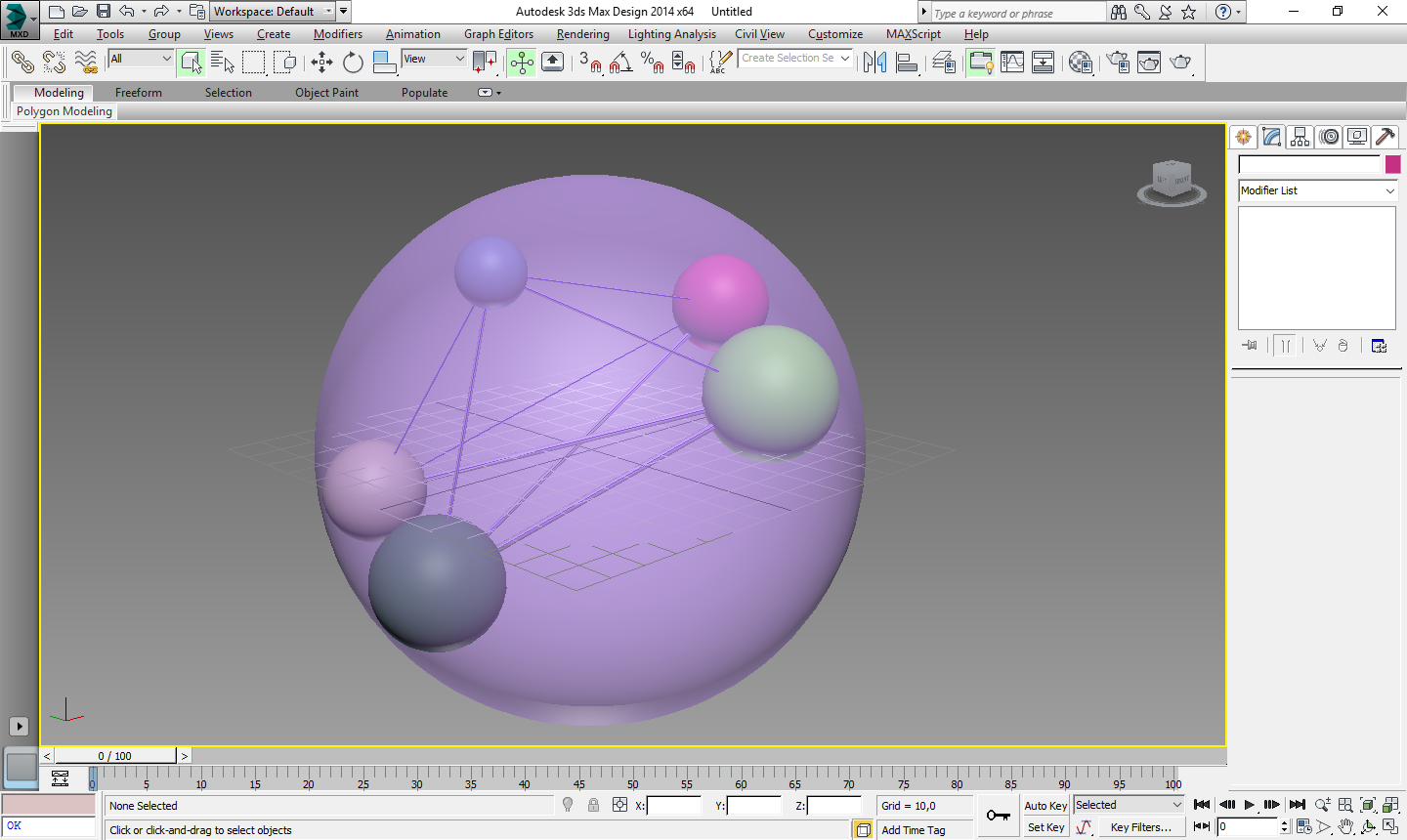


Рисунок 3.54 – Качество решения для 5 шаров с максимально-допустимым радиусом в 30 условных единиц

Запускаем солвер для максимального радиуса 30 на 7-ми шарах (рис. 3.55):

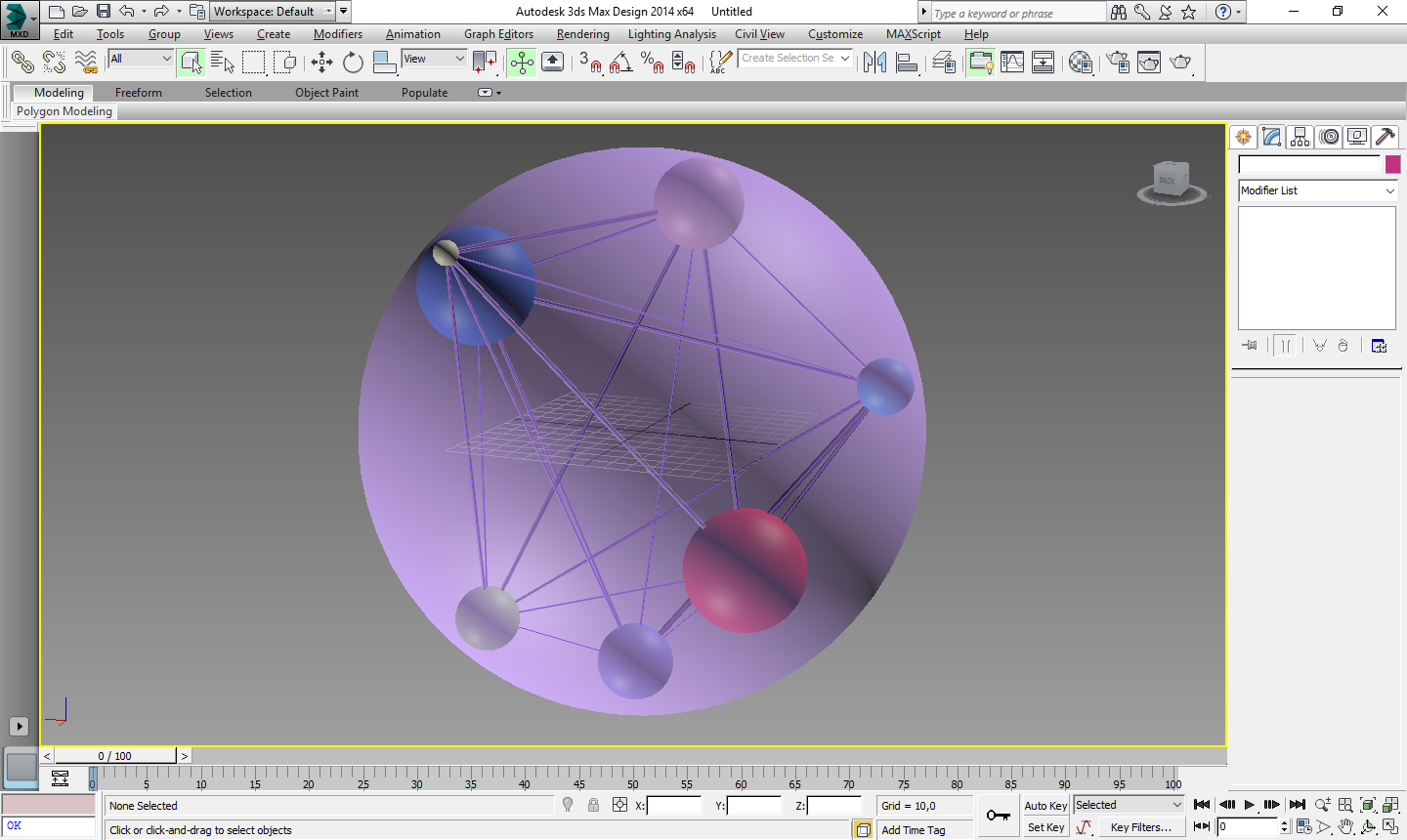


Рисунок 3.55 – Время счета для 7 шаров с максимальным радиусом в 40 единиц

Покажем качество решения на рисунке 3.56 для 18 шаров с максимальным радиусом 30 единиц:

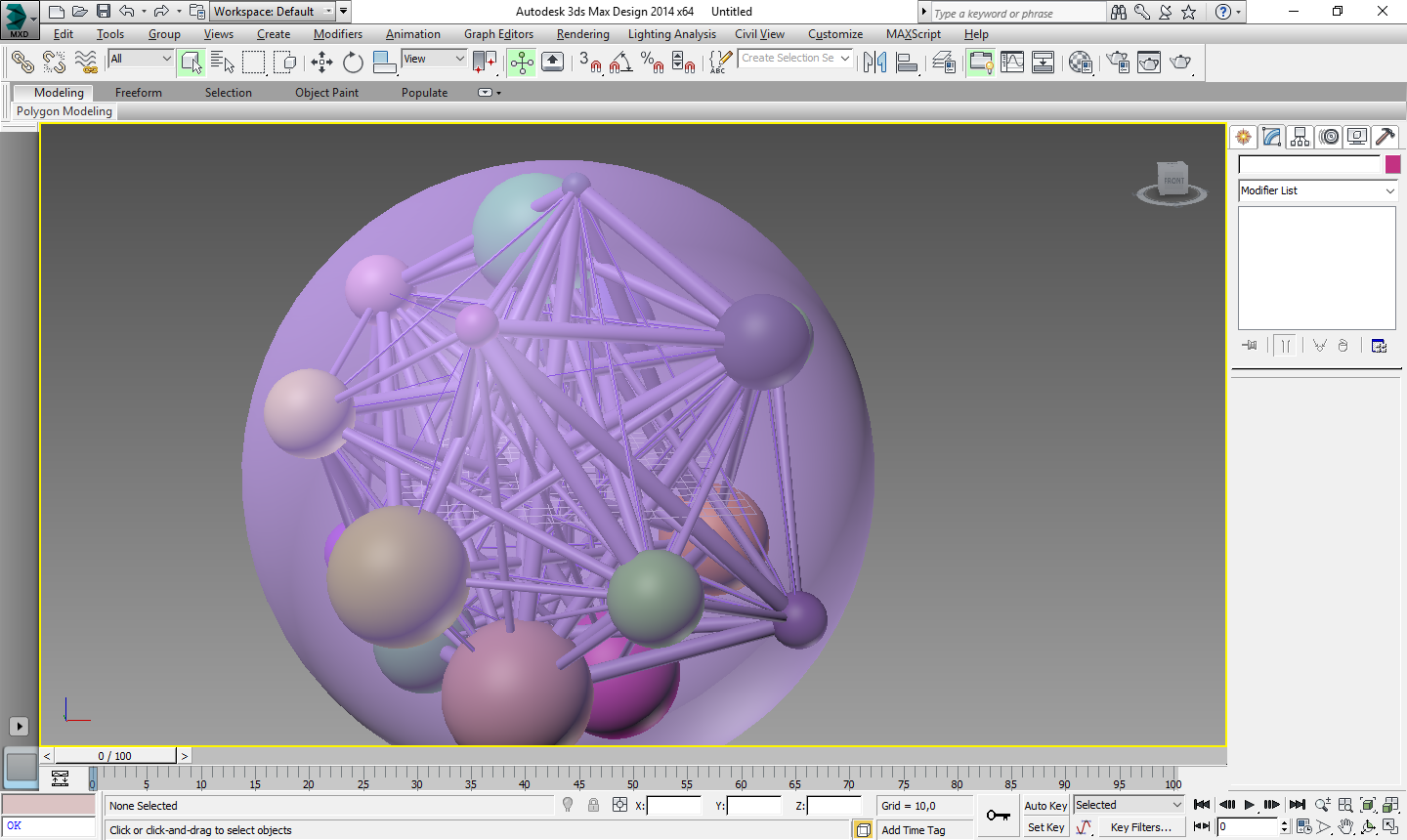


Рисунок 3.56 – Качество решения для 18 шаров с максимально радиусом в 30 условных единиц

После всех экспериментов перейдем к заключительным словам.

## Выводы к разделу 3

В этом разделе было сформировано полное детальное руководство пользователя, которое пошагово описывает работу с программным продуктом, решающим задачу равновесной упаковки объектов в заданной области по критерию минимизации длины связывающей сети.

Было произведен обзор документации к солверу IPOPT, по которой был подготовлен и сформирован формат передачи данных в решатель. Было построена диаграмма взаимодействия, которая дает четкое понимание программы на концептуальном уровне, дает возможность определить требований к системе, помогает пользователю выполнять свои задачи при работе с ПО.

Была рассмотрена концепция ООП, с применением которой и производилось проектирования программы. Были рассмотрены главные принципы ООП, было отобрано наиболее подходящие к постановке задачи.

Было произведено замеры скорости поиска локально-оптимального решения задачи равновесной упаковки шаров в контейнер с применением критерия минимизации длины связующей сети в зависимости от количества компонуемых объектов.

Было выяснено экспериментальным путем, что полученное решение можно пытаться улучшить путем передачи полученного локального решения в качестве начальной точки для входа в программу через текстовый файл, заполненный вручную или сформированный автоматически при помощи программы-скрипта в 3ds MAX, такие попытки не всегда приносят какие-либо изменения в качество искомого решения из-за, уже достигнутой, на предыдущей итерации, некой локально-оптимальной точки.

# ВЫВОДЫ

В ходе написания дипломной работы было подобрано список литературы по теме «Равновесная упаковка объектов в заданной области по критерию минимизации длины связывающей сети» относительно которой было описано входящие данные для исходной системы. По изученной теории методов оптимизации было построено математическую модель.

Было произведено качественный и сравнительный анализ существующих платных и бесплатных пакетов или встраиваемых библиотек в программный код с определенным набором алгоритмов для решения различных задач оптимизации. В результате обзора существующих пакетов, был выбран IPOPT, как пакет реализовывающий прогрессивный метод внутренней точки для поиска локального минимума и позволяющий работать с невероятно огромной размерности задачами и дающий достаточно быструю сходимость к решению с заданной машинной точность, в сравнении с градиентными методами. Был описан поиск глобального минимума. Был рассмотрен алгоритм поиска локального экстремума методом внутренней точки. Было получено количество итераций до схождения к решению с указанной точностью в лучшем и худшем случаях.

При проектировании программы было предусмотрено концепцию ООП. Было предусмотрено приведение данных к специальному виду перед непосредственной передачей в солвер IPOPT.

В работе были проведен анализ эмпирических зависимостей времени счета в задачах компоновочного синтеза круговых объектов с учетом критерия минимально длины связи и ограничений попарного не пересечения, попадания в область контейнера, состояния небаланса. В соответствии были получены кривые зависимостей временных характеристик от количества компонуемых шаровых объектов. Для проверки качества решения

Было выявлено, что, для некоторых ситуаций можно пытаться улучшить полученное решение путем задания в качестве начальной точки полученное локально-оптимальное решение на предыдущей итерации программы.

# СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Стоян, Ю. Г. Математические модели и оптимизационные методы геометрического проектирования [Текст] : учеб. / Ю. Г. Стоян, С. В. Яковлев. – К. : Академия Наук УССР Институт проблем машиностроения, 1986. – С. 202–227.

Pichugina, O. S. Continuous Representations and Functional Extensions in Combinatorial Optimization [Текст] : учеб. / O. S. Pichugina, S. V. Yakovlev. – M. : Cybernetics and Systems Analysis, 2016. – С. 921–930.

Fasano, G. Optimized Packings and Their Applications [Текст] : учеб. / G. Fasano, J. D. Pintér. – М. : Springer Opt. and its Apply, 2015. – 105 p.

Stetsyuk, P. I. On the global minimum in a balanced circular packing problem [Текст] : учеб. / P. I. Stetsyuk, T. E. Romanova. – М. : Springer Opt. and its Appl. 2015. – 105 с.

Stoyan, Yu. G. Packing Unequal Spheres into Various Containers [Текст] : учеб. / Yu. G. Stoyan, G. Scheithauer, G. N. Yaskov – M. : Cybernetics and Systems Analysis, 2016. – 419 p.

Che, C. Test problems for quasi-satellite packing: cylinders packing with behavior constraints and all the optimal solutions known [Текст] : учеб. / C. Che, Y. Wang, H. – T. : Teng Optimization Online, 2008. – 340 p.

Fasano, G. Modeling and Optimization in Space Engineering. Series: Springer Optimization and Its Applications [Текст] : учеб. / G. Fasano. – H. : Springer Optimization and Applications, 2013. – 404 p.

Stetsyuk, P. I. On the global minimum in a balanced circular packing problem [Текст] : учеб. / P. I. Stetsyuk, T. E. Romanova, G. Scheithauer. – T. : Optimization Letters, 2015. – 1347 p.

Stoyan, Yu. Packing unequal circles into a strip of minimal length with a jump algorithm [Текст] : учеб. / Yu. Stoyan, G. Yaskov. – T. : Optimization Letters, 2014. – 949 p.

Fasano, G. Modeling and Optimization in Space Engineering. [Текст] : учеб. / G. Fasano, J. D. Pintér. – М. : Springer Opt., 2013. – 404 p.

Stetsyuk, P. I., On the global minimum in a balanced circular packing problem [Текст] : учеб. / P. I. Stetsyuk, T. E. Romanova – .M : Optimization Letters, 2015. – P. 1347–1360.

Stoyan, Yu., Packing unequal circles into a strip of minimal length with a jump algorithm [Текст] : учеб. / Yu. Stoyan, G. Yaskov.. – М. : Optimization Letters. 2014. – 860 с.

Yakovlev, S. V. The method of artificial space dilation in problems of optimal packing of geometric object [Текст] : учеб. / S. V. Yakovlev. — K.: Cybernetics and Systems Analysis, 2017. – С. 725–732.

Яковлев, С. В. Метод переменных радиусов в задаче размещения неравных кругов [Текст] : статья / С. В. Яковлев, О. В. Карташов. — Х.: Наук. думка, 2017. – С. 1–5.

Яковлев, С. В. О комбинаторной структуре задач оптимального размещения геометрических объектов [Текст] : докл. / С. В. Яковлев. — Х.: Доклады НАН Украины 2017. – №9. – С. 26–30.

Стоян, Ю. Г. Решение некоторых многоэкстремальных задач методом сужающихся окрестностей [Текст] : статья / Ю. Г. Стоян, В. З. Соколовский. – К.: Наук. думка, 1980. – 256 с.

Емеличев, В. А. Многогранники, графы, оптимизация (комбинаторная теория многогранников) [Текст] : учеб. пособие / В. А. Емеличев, М. М. Ковалев, М. К. Кравцов. – М. : Наука, 1981. – 344 с.

Шевченко, В. Н. Линейное и целочисленное программирование [Текст] : учеб. Пособие / В. Н. Шевченко, Н. Ю. Золотых. – Нижний Новгород : Университет, 2004. – 154 с.

Методы линейного программирования [Электронный ресурс] / Воронеж. Институт Менеджмента, Маркетинга и Финансов – Режим доступа : \www/ URL: http://math.immf.ru/lections/302.html/ – 08.12.2016 г. – Загл. с экрана.

Романова, Т. Е. Phi-функции для моделирования ограничений включения в оптимизационных задачах компоновки [Текст] : статья / Т. Е. Романова, А. А. Коваленко. – К.: Системы обработки информации, 2013. , № 117. – C. 228–133.

Stoyan, Yu. G. Mathematical models of placement optimization: Two- and three-dimensional problems and applications [Текст] : статья / Yu. G. Stoyan, G. Fasano. – P. : Space Engineering Center, – 2013. – P. 363 – 388.

Коваленко, А. А. Упаковка круговых цилиндров в цилиндрический контейнер с учетом специальных ограничений поведения системы [Текст]: журнал / А. А. Коваленко, А. В. Панкратов. – М., – 2013. – № 1. – С. 126–134.

Поляк, Б. Т. Метод Ньютона и его роль в оптимизации и вычислительной математике [Текст]: журнал / Б. Т. Поляк. – М., 2006. – 19 с.

Нестеров, Ю. Е. Методы выпуклой оптимизации [Текст]: пособие / Ю. Е. Нестеров. – М. : МЦНМО, 2010. – 281 с.

Нестерова, Ю. Е. Алгоритмическая выпуклая оптимизация [Текст]: уч. пособие / Ю. Е. Нестеров. – К., 2013. – 367 с.