Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«Южно-Уральский Государственный университет

(национально исследовательский университет)»

Филиал ФГАОУ ВО «ЮУрГУ (НИУ)» в г. Златоусте

Факультет «Техники и технологии»

Кафедра «Математика и вычислительная техника»

Симуляция искусственной жизни

Научно исследовательская работа

Руководитель доцент, к.т.н

Соколова Е.В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2019г.

Автор проекта   
студент группы ФТТ-307

Б.А. Мурашов

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2019г.

Проект защищен   
с оценкой

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2019г.

Златоуст 2019г

# **АННОТАЦИЯ**

Мурашов Б.А. Симуляция искусственной жизни. – Златоуст: ЮУрГУ, МиВТ; 2019 г.,13 стр., 4 иллюстраций, библиографический список 5 ссылок.

Доработка приложения, которое осуществляет и поддерживает механизм естественного отбора в созданной среде на движке Unity.

В работе производится анализ и доработка недочетов предыдущий версии. В конце приводится анализ эксперементальных данных, полученных в ходе работы программы.

**СОДЕРЖАНИЕ**

[АННОТАЦИЯ 3](#_Toc9886254)

[ПОСТАНОВКА ЦЕЛИ И ЗАДАЧИ 4](#_Toc9886255)

[ОПТИМИЗАЦИЯ ПРИЛОЖЕНИЯ 5](#_Toc9886256)

[РЕАЛИЗАЦИЯ СБОРА СТАТИСТИЧЕСКИХ ДАННЫХ 8](#_Toc9886257)

[АНАЛИЗ СТАТИСТИЧЕСКИХ ДАННЫХ 10](#_Toc9886258)

[ВЫВОД 12](#_Toc9886259)

[БИБЛИОГРАФИЧЕСКИЙ СПИСОК 13](#_Toc9886260)

# **ПОСТАНОВКА ЦЕЛИ И ЗАДАЧИ**

Целью исследовательской работы является доработка разработанной версии программы.

Задачи проекта:

1. провести оптимизацию приложения;
2. добавить метод сбора статистических данных;
3. провести анализ данных.

# **ОПТИМИЗАЦИЯ ПРИЛОЖЕНИЯ**

В качестве примера разберем оптимизацию метода движения бота MoveForward.

Код до оптимизации:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class MoveForward : MonoBehaviour {

/// <summary>

/// object reference with characteristics

/// </summary>

[SerializeField] Specifications Stat;

private float \_distance;

/// <summary>

/// Move bot forward

/// </summary>

public void Move() {

\_distance = 0;

///Check border field

if (490 <= Stat.Speed + transform.position.x || -490 >= Stat.Speed + transform.position.x || 490 <= Stat.Speed + transform.position.z || -490 >= Stat.Speed + transform.position.z) {

///Fine network

Stat.Web.ValueCrit--;

Stat.HP -= 10;

return;

}

StartCoroutine( MoveAnimation() );

}

/// <summary>

/// Animation parallel flow

/// </summary>

IEnumerator MoveAnimation() {

if (\_distance < Stat.Speed) {

if (490 <= transform.position.x || -490 >= transform.position.x || 490 <= transform.position.z || -490 >= transform.position.z) {

StopCoroutine( MoveAnimation() );

yield return null;

}

else {

\_distance += Stat.Speed / 20f;

transform.Translate( Stat.Speed / 20f, 0, 0 );

yield return new WaitForSeconds( 0.01f );

StartCoroutine( MoveAnimation() );

}

}

else {

StopCoroutine( MoveAnimation() );

}

}

}

Код после оптимизации:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class MoveForward : MonoBehaviour {

/// <summary>

/// ссылка на объект с характеристиками

/// </summary>

[SerializeField] Specifications Stat;

/// <summary>

/// Триггер разрешения движения

/// </summary>

private bool \_enter = true;

/// <summary>

/// Пройденное расстояние

/// </summary>

private float \_distance;

/// <summary>

/// Move bot forward

/// </summary>

public void Move() {

\_distance = 0;

///Проверка возможности движения

if ( !\_enter ) {

///Штраф к нейронной сети

Stat.Web.ValueCrit--;

Stat.HP -= 10;

return;

}

else {

Stat.BotData.Move++;

StartCoroutine ( MoveAnimation ( ) );

}

}

/// <summary>

/// Анимация в параллельных потокахы

/// </summary>

IEnumerator MoveAnimation () {

if (\_distance < Stat.Speed) {

if (!\_enter) {

StopCoroutine( MoveAnimation() );

yield return null;

}

else {

\_distance += Stat.Speed / 20f;

transform.Translate( Stat.Speed / 20f, 0, 0 );

yield return new WaitForSeconds( 0.01f );

StartCoroutine( MoveAnimation() );

}

}

else {

StopCoroutine( MoveAnimation() );

}

}

/// Проверка пересечения границы с другим объектом

private void OnTriggerEnter(Collider collision) {

if (collision.gameObject.tag == "Wall" || collision.gameObject.tag == "Bot" || collision.gameObject.tag == "Organic" )

\_enter = false;

}

private void OnTriggerExit(Collider collision) {

if (collision.gameObject.tag == "Wall" || collision.gameObject.tag == "Bot" || collision.gameObject.tag == "Organic" )

\_enter = true;

}

}

Условие выхода замененно на реации на события «соприкосновение» с другими объетами, в результате которых происходит переключение триггера, разрешающего движение. Проверка на пересечения границ не происходит каждый вызов метода.

# **РЕАЛИЗАЦИЯ СБОРА СТАТИСТИЧЕСКИХ ДАННЫХ**

В скрипт смерти бота дописан метод, отвечающий за сохранения статистики лучшего представителя поколения, в случаи смерти всего поколения.

private void SaveData ( ) {

StreamWriter objFile;

if ( !File.Exists ( "Log.txt" ) )

File.Create ( "Log.txt" );

objFile = new StreamWriter ( "Log.txt", true );

objFile.WriteLine ( Stat.Name + "\_" + Stat.LifeTime.ToString ( ) + "\_" + Stat.BotData.Move.ToString ( ) + "\_" + Stat.BotData.Kill.ToString ( ) + "\_" + Stat.BotData.Eat.ToString ( ) + "\_" + Stat.BotData.Generation.ToString ( ) );

objFile.Close ( );

Debug.Log ( "SaveData" );

}

В класс Specifications, который хранит параметры бота дописана структура, которая хранит статистические данные.

///<summary>Структура для статистики </summary>

public struct Statistic {

public int Move;

public int Kill;

public int Eat;

public int Generation;

}

В результате работы генерируется файл, содержащий статистические данные вида:

G6095L\_46\_45\_0\_0\_0

M5979S\_63\_63\_0\_0\_0

Q2095D\_46\_46\_0\_0\_0

Q9710K\_78\_67\_0\_11\_0

O2088Z\_46\_1\_0\_45\_0

A3915D\_46\_41\_0\_0\_0

N7332Y\_47\_0\_0\_47\_0

G4506A\_47\_47\_0\_0\_0

R5831F\_46\_27\_0\_17\_0

V5897Y\_46\_0\_0\_46\_0

E9060M\_46\_0\_0\_40\_0

U7615D\_45\_45\_0\_0\_0

B5066W\_46\_46\_0\_0\_0

Y7633T\_46\_1\_0\_41\_0

L1910V\_12\_8\_0\_0\_0

B9689U\_46\_40\_0\_1\_0

U7700H\_46\_41\_0\_0\_0

X4548Q\_46\_0\_0\_46\_0

Y7744V\_24\_24\_0\_0\_0

A4893U\_47\_45\_0\_2\_0

L6708A\_47\_0\_0\_47\_0

# **АНАЛИЗ СТАТИСТИЧЕСКИХ ДАННЫХ**

В результате 10000 поколений полученны эксперементальные данные в том же колличестве. Они включают: имя, возрас, колличество различных действй бота.

Экспортируя данные в Excel, получаем таблицу, часть которой представлена на рисунке 1.

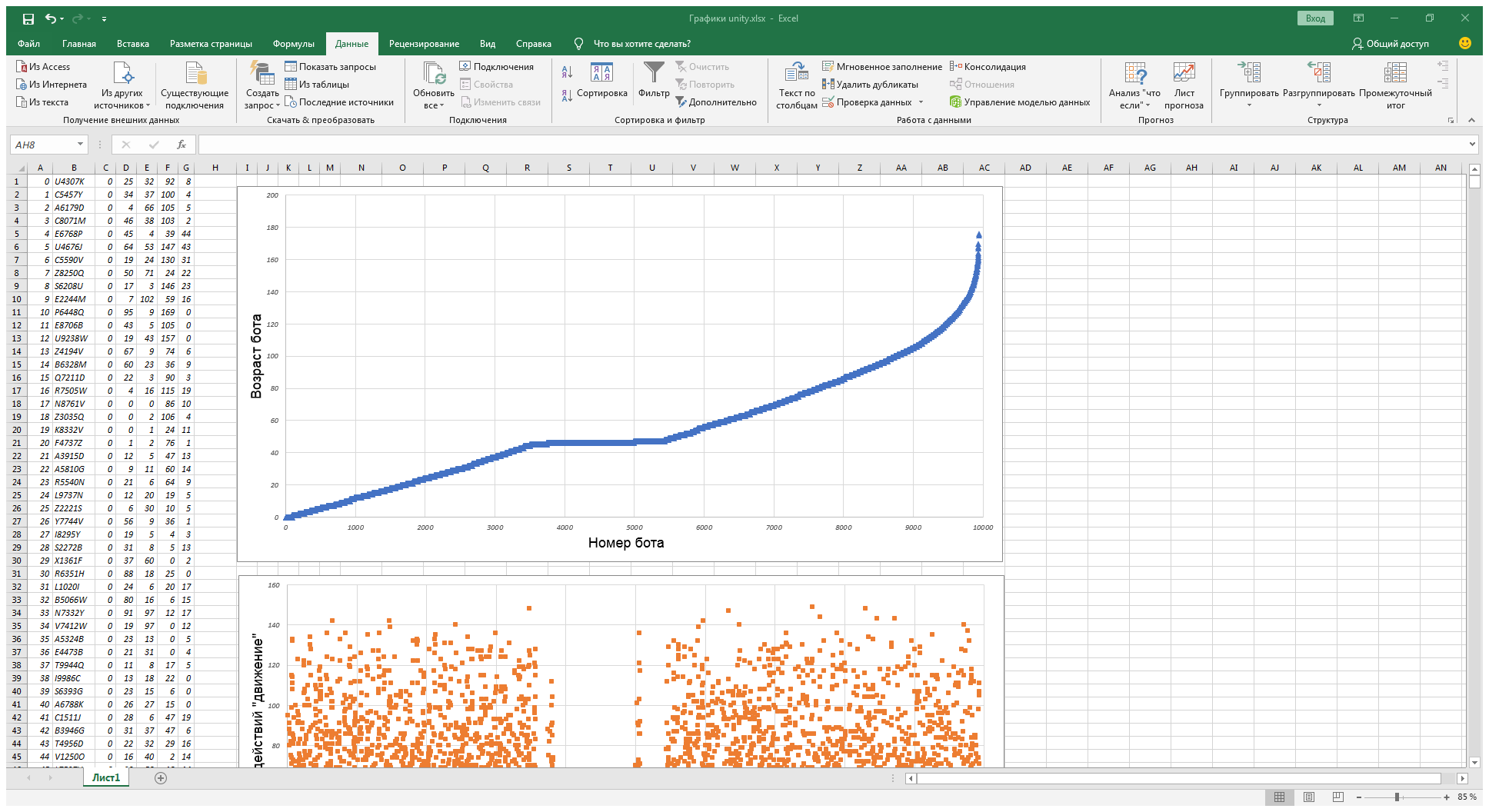


Рисунок 1 – Эксперементальные данные

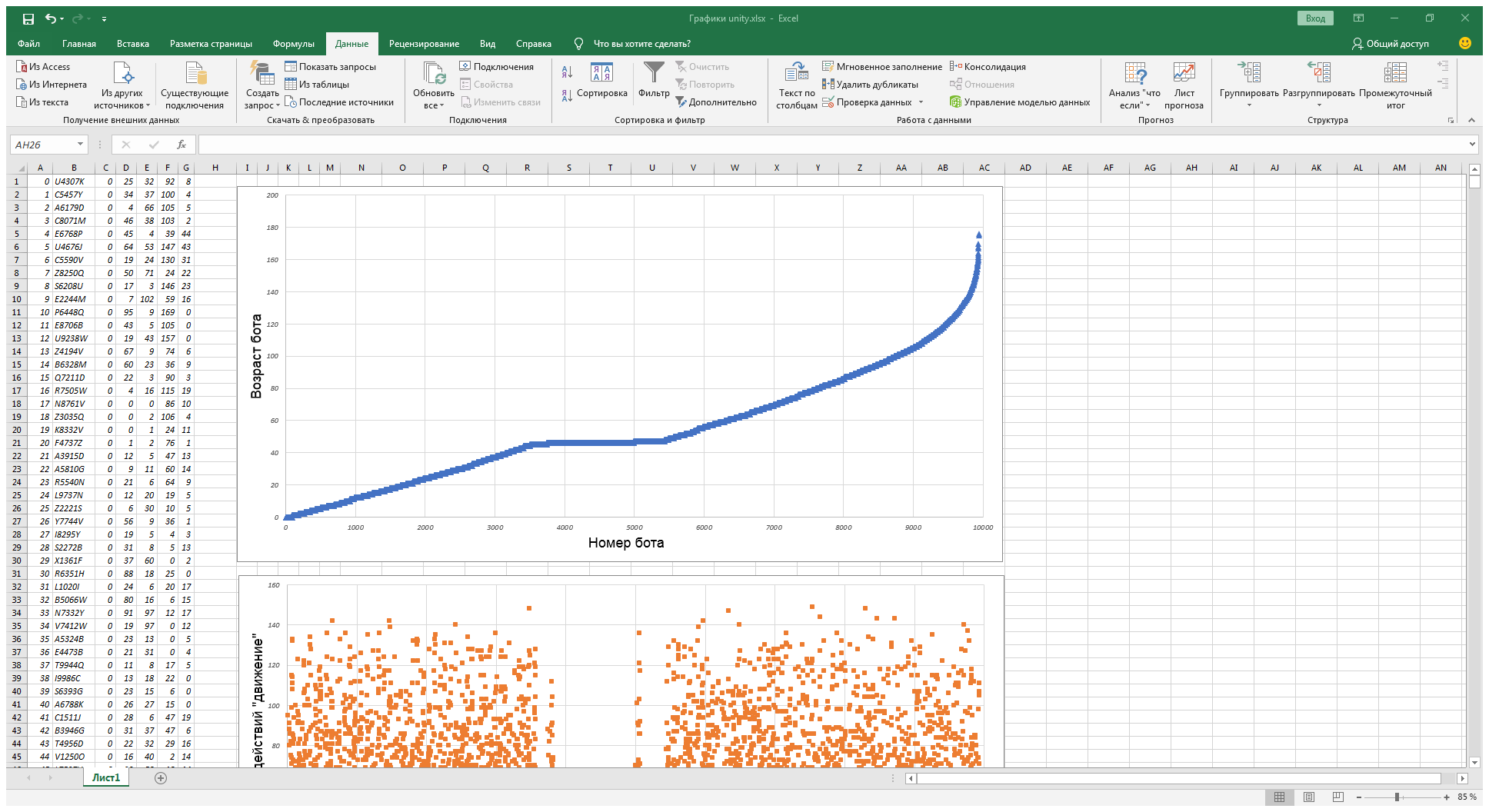
Составим графики по полученным данным (рисунок 2, 3, 4).

Рисунок 2 – График зависимости возраста от номера расчета

По графику видно, что в отличии от предедущий версии программы, прогресс идет быстрее и более резко. Это обусловнено новой схемой нейроной сети, метода мутации и критерий отбора лучшего экземпляра нейронной сети.

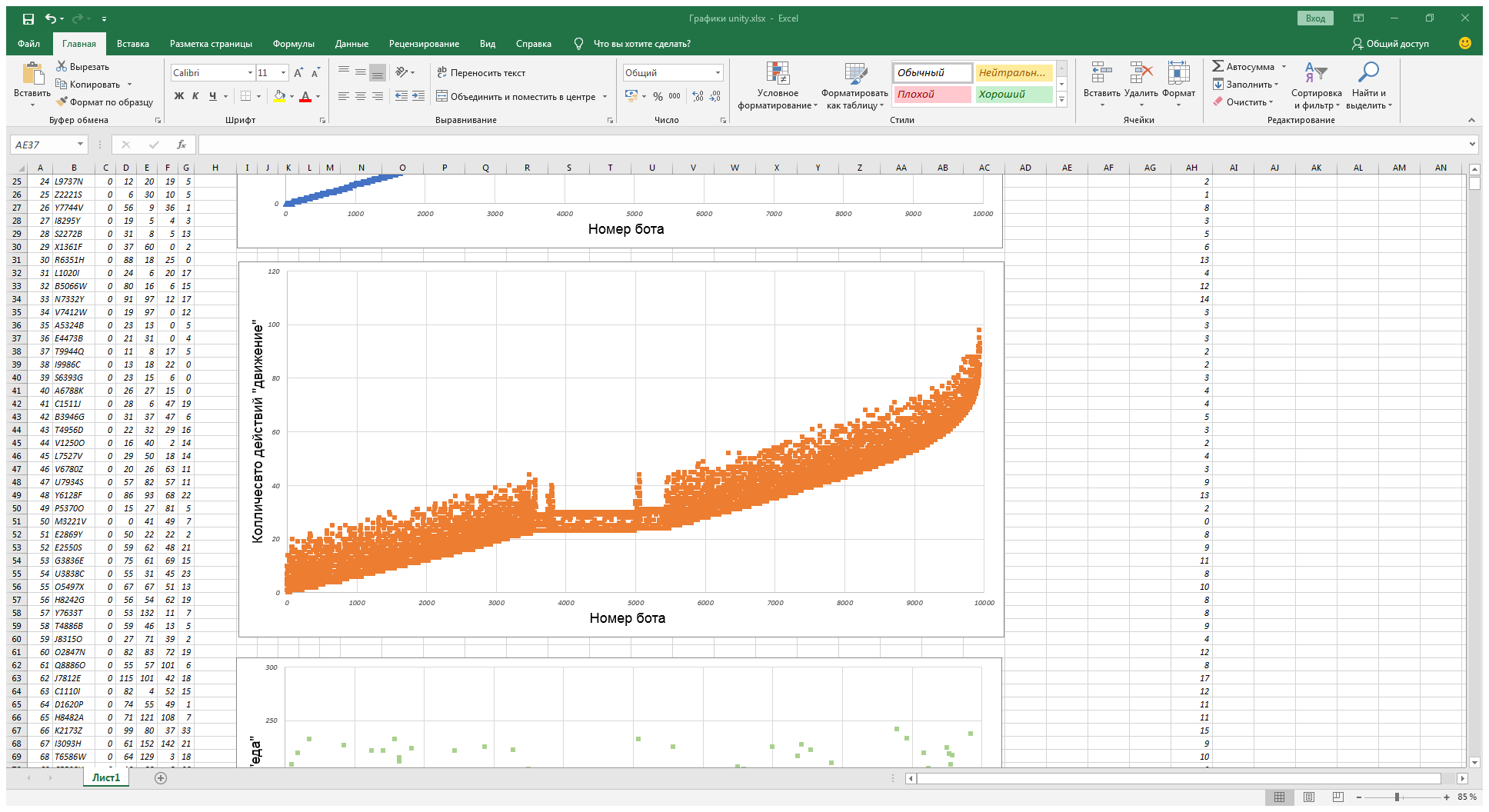
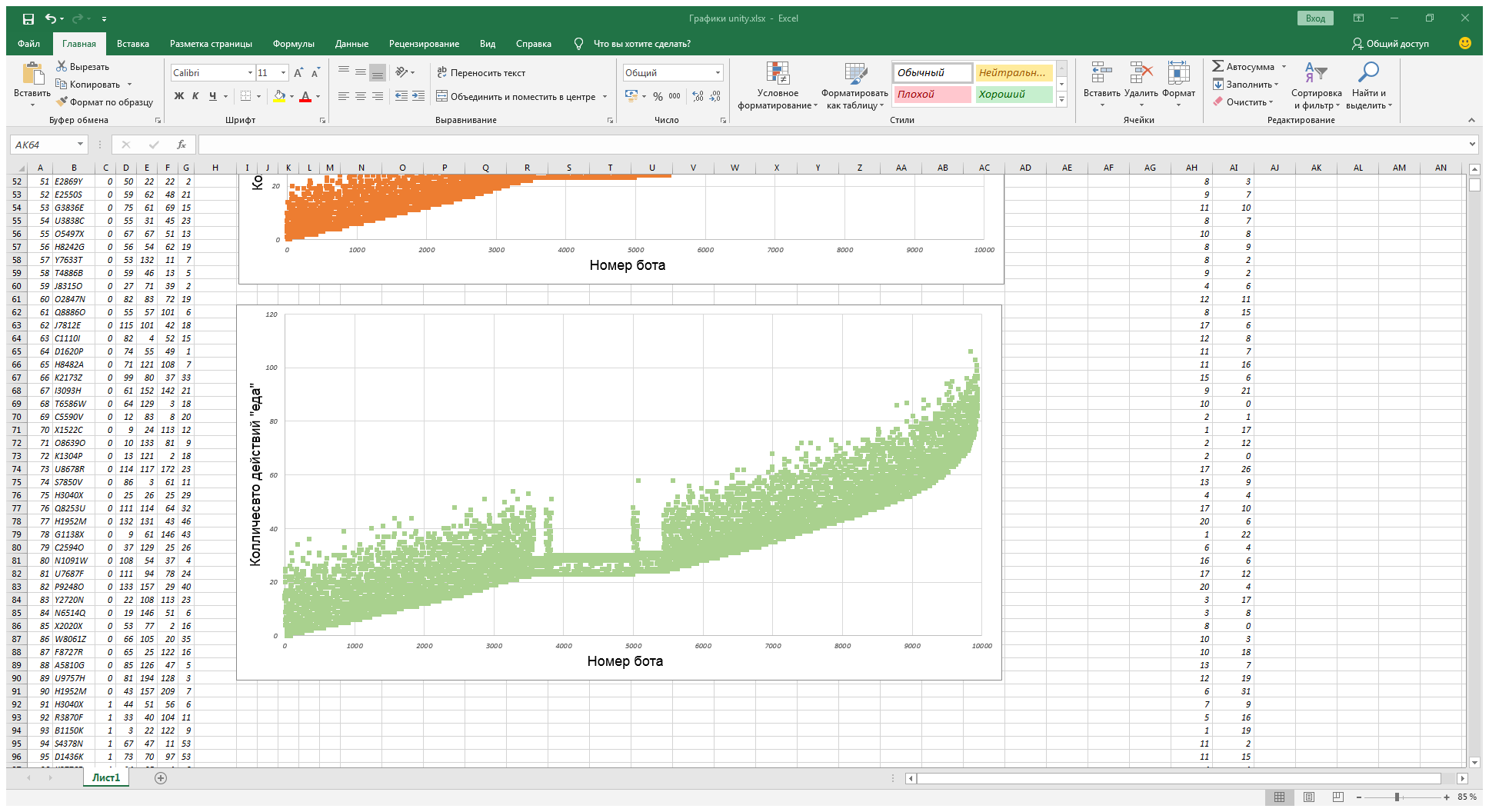
Рисунок 3 – График зависимости номера расчета от количества действий «движение»

Рисунок 4 - График зависимости номера расчета от количества действий «еда»

Исходя из полученных графиков, можно предположить, что колличесвто движений и действий еда имеют прямую зависимоть с продолжительностью жизни. Из-за системы рейтинга нейронных сетей, лишнии движения отсекатся, значит боты совершили столь много движений из необходимости.

# **ВЫВОД**

Подводя итоги, можно сказать: выполненая оптимизация увеличила скорость работы приложения, использование unity позволяет создать масштабируемую систему, увеличивая эффекцтивность приложения. Статистические данные, в сравнение с данными из первой версии приложения указывают на более эфективную архитектуру нейронной сети, пользу системы рейтингов и методов хранения нейронных сетей.

# **БИБЛИОГРАФИЧЕСКИЙ СПИСОК**

* Андрейчиков А. В., Андрейчикова О. Н. Системный анализ и синтез стратегических решений в инноватике. Математические, эвристические и интеллектуальные методы системного анализа и синтеза инноваций. Учебное пособие; Ленанд - М., 2015. - 306 c.
* Редько В. Г. Моделирование когнитивной эволюции. На пути к теории эволюционного происхождения мышления; Ленанд - М., 2015. - 256 c.
* Тархов Д. А. Нейронные сети. Модели и алгоритмы. Книга 18; Радиотехника - М., 2012. - 256 c.
* Люгер Дж. Ф. Искусственный интеллект: стратегии и методы решения сложных проблем = Artificial Intelligence: Structures and Strategies for Complex Problem Solving / Люгер Дж. Ф., Под ред. Н. Н. Куссуль. - 4-е изд. - Москва: Вильямс, 2005. - 864 с.
* И. Рузмайкиной. Unity в действии. Мультиплатформенная разработка на C#/ Пер. с англ. – СПБ.: Питер, 2016. – 336 с.: ил. – (Серия «Для профессионалов»).