Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Информационная безопасность

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОМУ ПРОЕКТУ

НА ТЕМУ:

Моделирование эволюции на основе генетических алгоритмов

Студент ИУ8-31 _____ Важенина А. В.
(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель курсового проекта _____ Бородин А. А.
(Подпись, дата) (И.О.Фамилия)

Консультант _____ _____
(Подпись, дата) (И.О.Фамилия)

2018 г.

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ
Заведующий кафедрой ИУ8
(Индекс)
Басараб М.А.
(И.О.Фамилия)
« _____ » _____ 20 ____ г.

З А Д А Н И Е
на выполнение курсового проекта

по дисциплине Технологии и методы программирования

Студент группы ____ИУ8-31_____

_____Важенина Анастасия Витальевна_____
(Фамилия, имя, отчество)

Тема курсового проекта __Моделирование эволюции на основе генетических алгоритмов___

Направленность КП: учебный
Источник тематики (кафедра, предприятие, НИР): кафедра

График выполнения проекта: 25% к 3 нед., 50% к 9 нед., 75% к 12 нед., 100% к 15 нед.

Задание:
Создать модель развития эволюции в альтернативном мире, которая позволяет увидеть и проанализировать естественный процесс развития живой природы, сопровождающийся изменением генетического состава популяций, формированием адаптаций, видообразованием и вымиранием видов, преобразованием экосистем и биосферы в целом. Модель должна четко отражать поведение живых организмов в представленной среде обитания, показывать преимущества и недостатки мутационных генов в развитии и адаптации вида, иметь гибкую архитектуру в случае умышленного изменения генома и иметь высокую производительность при большом количестве организмов
Оформление курсового проекта:

Расчетно-пояснительная записка на _____ листах формата А4.
Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)
_____
_____
_____
_____

Дата выдачи задания « 11 » сентября 2018 г.

Руководитель курсового проекта _____ __Бородин А. А._____
(Подпись, дата) (И.О.Фамилия)

Студент _____ __Важенина А. В._____
(Подпись, дата) (И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

Оглавление

Цель

Цель проекта заключается в создании модели развития эволюции в альтернативном мире, которая позволяет увидеть и проанализировать естественный процесс развития живой природы, сопровождающийся изменением генетического состава популяций, формированием адаптаций, видообразованием и вымиранием видов и тому подобное.

Основные определения

- Evolution game — название проекта.
- SFML — это простая и кроссплатформенная мультимедиа библиотека. SFML обеспечивает простой интерфейс для разработки игр и прочих мультимедийных приложений.
- ПО — программное обеспечение.
- Поколение — общность каких-то объектов (людей, животных, растений, иногда даже неодушевлённых предметов) по длине цепи непосредственных предков до некоторого родоначальника (группы таковых); или же по времени рождения.
- Геном — совокупность наследственного материала, заключенного в клетке организма. Геном содержит биологическую информацию, необходимую для построения и поддержания организма. (здесь: характеристика организма)
- Искусственная нейронная сеть(ИНС) — математическая модель, а также её программное или аппаратное воплощение, построенная по принципу организации и функционирования биологических нейронных сетей— сетей нервных клеток живого организма.
- Хромосома — здесь: «клетка» для запоминания генетической информации.
- Инициализация — это создание, активация, подготовка к работе, определение параметров. Приведение программы или устройства в состояние готовности к использованию.
- Оценка — дает возможность определить, как каждая хромосома (решение) справляется с данной проблемой.
- Отбор (или селекция) — на этом этапе хромосомы выбираются для дальнейшего использования в другой популяции.
- Рекомбинация — это перераспределение генетической информации путём физического обмена участками хромосом.

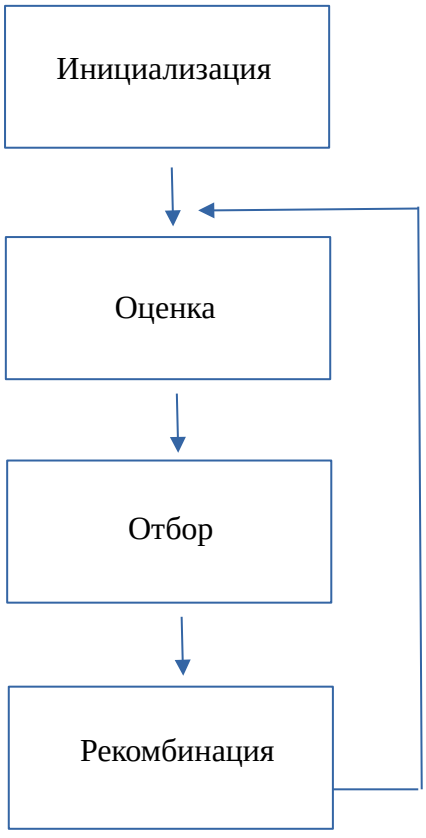
Введение

Проект показывает упрощенную модель эволюции, в основе который лежит реализация генетического алгоритма, соединенного с искусственной нейронной сетью.

Генетический алгоритм представляет собой технику оптимизации, которая моделирует феномен естественной эволюции (впервые открытый Чарльзом Дарвином). При естественной эволюции выживают и дают самое многочисленное потомство особи, наиболее адаптированные к сложным условиям окружающей среды. Степень адаптации, в свою очередь, зависит от набора хромосом конкретной особи, полученным от родителей. Это основа выживания сильнейшего — не только процесс выживания, но и участие в формировании следующего поколения. В природе выживание является определяющей и основной функцией.

Генетический алгоритм не пытается оптимизировать единственное решение. Он работает с группой решений, которые кодируются, подобно хромосомам. Отдельные гены хромосомы представляют собой уникальные переменные для изучаемой проблемы.

Выполнение генетического алгоритма включает три основных шага (причем для каждого предусмотрен большой разброс возможных вариантов):



Генетический алгоритм выполняется в три этапа (если не учитывать начальное создание популяции). Во время оценки определяется здоровье популяции. Далее производится отбор подгруппы хромосом на основании предварительно заданного критерия. Наконец, выбранная подгруппа рекомбинируется, в результате чего получается новая популяция. Алгоритм выполняется заново с новой популяцией. Процесс продолжается до тех пор, пока не будет достигнут определенный предел. Тогда работа алгоритма считается завершенной.

В проекте в качестве генома выступает искусственная нейронная сеть, веса которой являются генами хромосомами, которые в процессе эволюции будут изменяться. ИНС позволит каждому организму принимать решения о его действиях в окружающей среде, в которой он живет.

Нейронные сети (Neural network) представляют собой упрощенную модель человеческого мозга. Мозг состоит из нейронов , которые являются индивидуальными процессорами. Нейроны соединяются друг с другом с помощью нервных окончаний двух типов: синапсов, через которые в ядро поступают сигналы, и аксонов, через которые нейрон передает сигнал далее. Человеческий мозг состоит примерно из 10^{11} нейронов. Каждый нейрон связан примерно с 1000 других нейронов. Структура мозга высокоциклична, но ее можно рассматривать и как многослойную. В очень упрощенном виде работу мозга можно представить так: внешний слой сети передает импульсы от сенсоров из внешней среды, средний слой (или кора головного мозга) обрабатывает импульсы, а «выходной» слой выдает результат (действие) обратно во внешнюю среду.

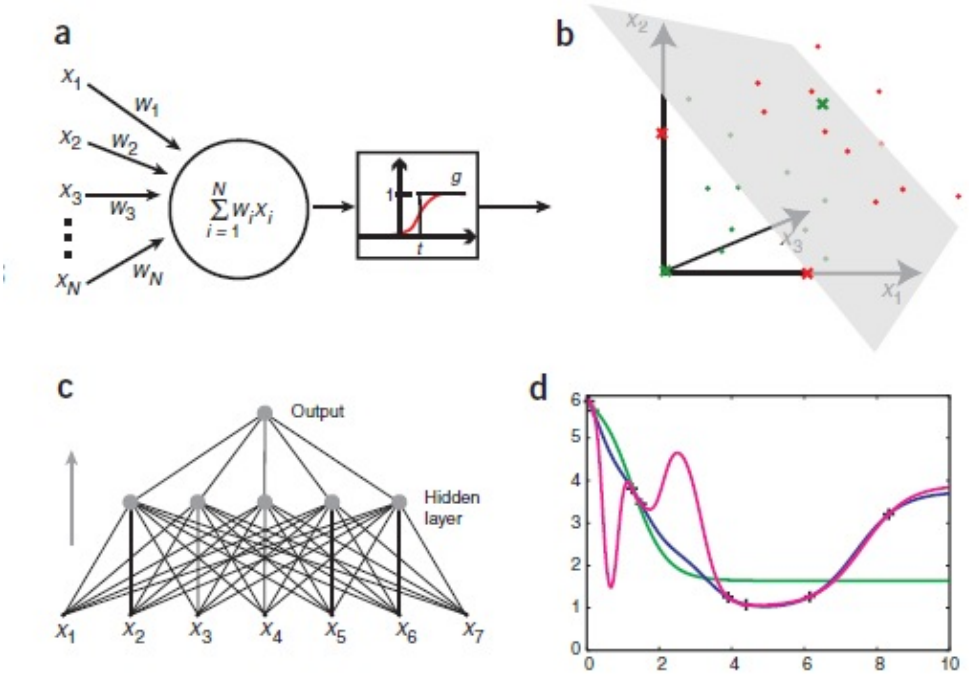
Искусственные нейронные сети имитируют работу мозга. Информация передается между нейронами, а структура и вес нервных окончаний определяют поведение сети.

ИНС создаются зачастую путем имитации модели сетей нейронов на компьютере. Мы можем заставить нейронную сеть «обучаться», используя алгоритмы, имитирующие процессы реальных нейронов головного мозга, что помогает решать различные проблемы различной сложности. Модель нейрона представляется как пороговая величина (1а). Такая модель получает

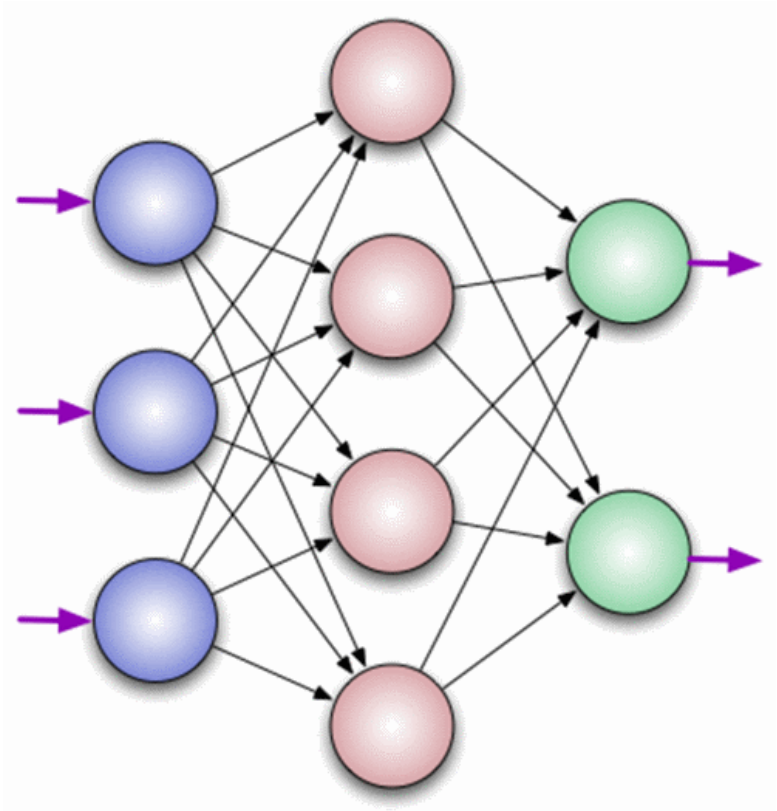
данные из внешних источников, после определяет значения каждого входа, а после добавляет эти значения. Определяется некоторая пороговая величина, после вычисляется общи вход, который может быть соответственно либо выше, либо ниже пороговой величины. Если значение выше, то выход блока равен единице, а в противном случае — нулю. Благодаря такому алгоритму значение выхода изменяется от 0 до 1, когда общая «взвешенная» сумма входов будет равна значению пороговой величины. Точки, расположенные в исходном пространстве и удовлетворяющие этому условию, определяют гиперплоскости. Если рассматривать гиперплоскость в двух измерениях, то она является линией, а если в трех измерениях, то нормальной (перпендикулярной) плоскостью. Точки, расположенные с одной стороны от нее классифицируются как 0, а точки с другой стороны — 1. Таким образом задача классификации может быть решена при использовании пороговой величины, при условии что два класса будут разделены гиперплоскостью. Такие проблемы называют линейно сепарабельными (1b).

1c — однонаправленная ИНС.

1d — переобучение.



Теперь, чтобы понять, как же работают нейронные сети, давайте взглянем на ее составляющие и их параметры.

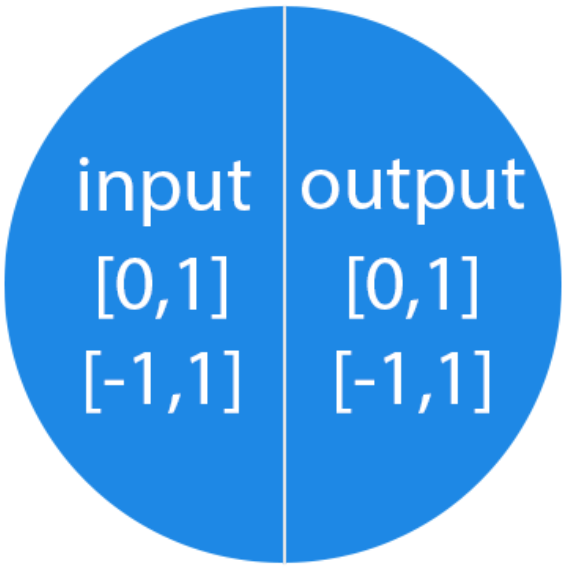


Начнем с определения нейрона: нейрон — это вычислительная единица, которая получает информацию, производит над ней простые вычисления и передает ее дальше.

Нейроны делятся на три основных типа:

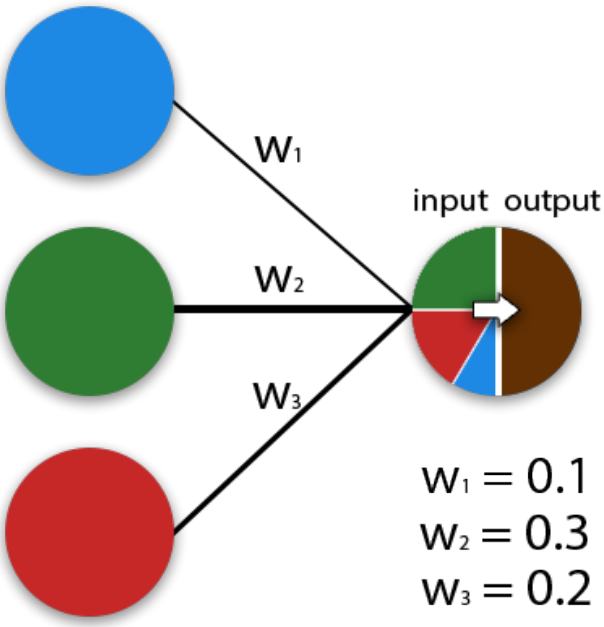
- входной нейрон (помечен синим цветом)
- скрытый нейрон (помечен красным цветом)
- выходной нейрон (помечен зеленым цветом)

Когда в сеть входит большое количество нейронов, вводят такое понятие как слой. Тогда нейроны, которые получают информацию, составляют входной слой, также есть n-ое количество скрытых слоев, которые обрабатывают информацию с входного слоя, и выходной слой, который выводит результат. У каждого нейрона есть два основных параметра: input data (входные данные) и output data (выходные данные). У входных нейронов input data = output data, у остальных нейронов input data является суммой всей информации с нейронов с предыдущего слоя, после она нормируется с помощью функции активации нейрона и переходит в параметр output data.

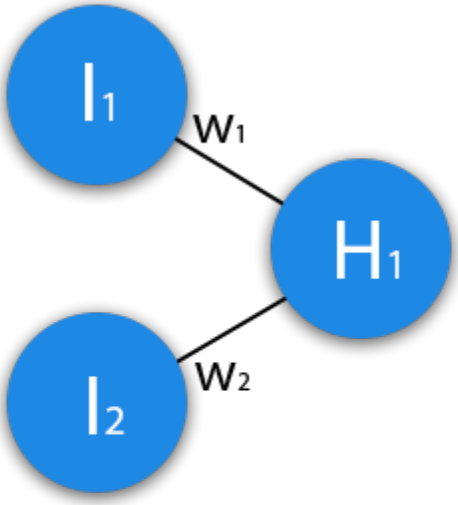


Дальше появляется вопрос: а что такое синапс?

Синапс — это связь между двумя нейронами. Сами синапсы имеют лишь один параметр — вес. Именно с помощью веса переданная информация меняется при передачи от одного нейрона к другому. Для примера рассмотрим такую ситуацию: у нас есть 3 нейрона, каждый передает свою информацию к следующему. Соответственно у нас есть 3 веса, которые соответствуют каждому из нейронов. Тот нейрон, у которого наибольший вес, будет передавать доминирующую информацию следующему нейрону. Совокупность весов нейронной сети или матрица весов — это своеобразный мозг всей системы. Именно благодаря этим весам, входная информация обрабатывается и превращается в результат.



Теперь рассмотрим как это все работает.



$$1) H_{1_{input}} = (I_1 * w_1) + (I_2 * w_2)$$

$$2) H_{1_{output}} = f_{activation}(H_{1_{input}})$$

Из формулы видно, что входная информация — это сумма всех входных данных, умноженных на соответствующие им веса. Тогда дадим на вход 1 и 0. Пусть w1=0.4 и w2 = 0.7 Входные данные нейрона H1 будут следующими: 1*0.4+0*0.7=0.4. Теперь когда у нас есть входные данные, мы можем получить выходные данные, подставив входное значение в функцию активации (подробнее о ней далее). Теперь, когда у нас есть выходные данные, мы передаем их дальше. И так, мы повторяем для всех слоев, пока не дойдем до выходного нейрона. Запустив такую сеть в первый раз мы увидим, что ответ далек от правильно, потому что сеть не натренирована. Для того чтобы улучшить результаты и нужна «тренировка» или обучение сети.

Выше уже проскальзывала фраза: функция активации нейрона. Теперь разберемся что это такое.

Функция активации — это способ нормализации входных данных. На вход приходят достаточно большие значения, которые не входят в диапазон допустимых значений, если мы пропускаем его через функцию активации, то результат не выходит за рамки, тем самым мы получаем результат в нужном нам диапазоне.

Теперь рассмотрим, как все описанные выше процессы происходят в проекте.

Пожалуй начнем с генетического алгоритма.

Инициализация происходит таким образом, что 10 организмов, которые составляют первое поколение, с весами нейронной сети (выступающими в роли хромосом), которые сгенерированы случайным образом, запускаются в сгенерированный окружающий и предоставлены сами себе. В этом мире для них есть еда, которая позволяет им дольше жить, и яд, который ведет их к гибели. Оценка организмов идет таким образом: собирается статистика «питания» с каждого организма, то есть сколько организм съел еды, а сколько яда. В конце каждого поколения идет анализ этой статистики. Концом поколения считается момент, когда последний организм из него умирает. После анализа данных производится отбор — селекция. По собранным данным отбираются организмы, которые за период своей жизни съели еды больше чем яда. Так мы отбираем особей, которые будут родителями следующего поколения. Рекомбинация происходит следующим образом: отобранные во время селекции особи клонируются и после о клонов — потомков — происходит мутация генов. Во время мутации выбираются от 1 до 3 весов во всей нейронной сети и они изменяются на небольшое число, которое будем назвать дельта.

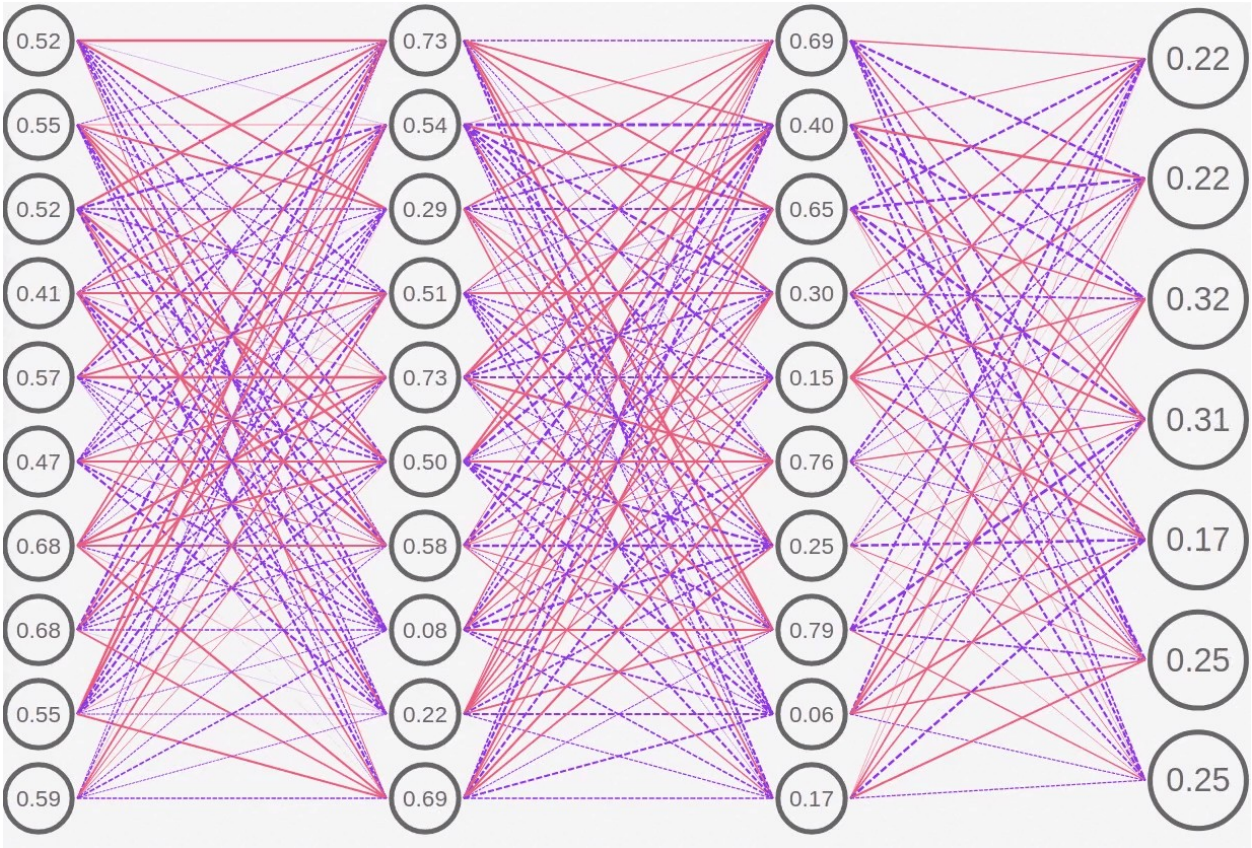
Весь алгоритм продолжается в каждом поколении тем самым происходит естественное обучение искусственной нейронной сети — происходит процесс эволюционирования.

Далее перейдем к работе искусственной нейронной сети в проекте.

Нейронная сеть получает на вход небольшой участок карты вокруг организма. Этот участок состоит из 6 шестиугольников (так как вся карта состоит из них), которые сеть должна обработать и принять решение о действии особи. ИНС состоит из 4 слоев: входного, 2-х скрытых слоев и выходного слоя. Значения всех нейронов изменяются от 0 до 1, а веса изменяются от -1 до 1. Входной слой состоит из 10 нейронов.

Рассмотрим их поподробнее:

1. Слой состоит из 10 нейронов, так как организм может быть повернут в одну сторону на каждой итерации и видеть перед собой только 3 клетки, оставшиеся 3, расположенные сзади, он не видит. Происходит 6 итераций — поворотов организма — чтобы собрать данные со всех 6 клеток вокруг организма. Для каждой видимой клетки есть три варианта: на этой клетке стоит еда, организм или яд.
2. Каждый первый нейрон (то есть N mod 3 = 1, где N номер нейрона) с 1 по 9 отвечает за расположение на клетке еды. Если там есть еда, тогда значение внутри нейрона будет равно 1, если нет — 0. Таким образом 1, 4 и 7 нейроны будут отвечать за расположение еды на трех клетках, которые видит организм. Каждый второй нейрон (N mod 3 = 2) согласно такой схеме будет отвечать за расположение на клетке другого организма. А каждый третий нейрон отвечает за расположение на клетке яда.
3. Самый последний десятый нейрон отвечает за решение двигаться организму или нет.



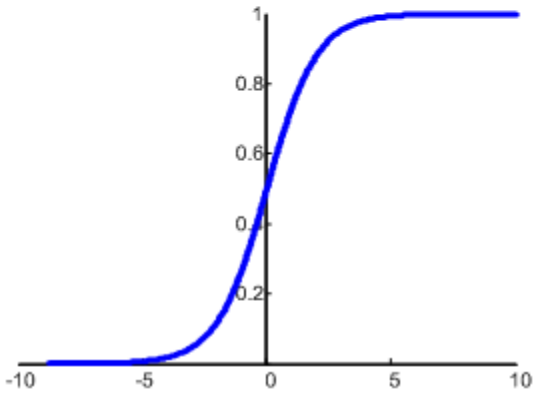
Также нейронная сеть имеет 2 скрытых слоя, которые состоят из 10 нейронов каждый.

Выходной слой имеет 7 нейронов: первые шесть нейронов отвечают за 6 клеток вокруг, а последний отвечает за решение идти или оставаться на месте. На каждой итерации данные собираются с трех клеток и самое большое значение решения суммируется со значением в соответствующей клетке, после того как пройдет 6 итераций — поворотов — каждая сумма пропускается через функцию активации и, выбирая самое большое значение, получаем решение нейронной сети.

Теперь рассмотрим поподробнее функцию активации, которая используется в проекте.

Здесь используется функция — сигмоид.

$$f(x) = \frac{1}{1+e^{-x}}$$



Сигмоид

является самой распространенной функцией активации, ее диапазон значений [0,1]. Именно эта функция является самой распространенной для примеров, также ее называют логистической функцией.

Требования к проекту

- Простое создание новых карт окружающего мира.
- Гибкая архитектура для изменения размеров карт и запуск на них организмов с произвольного поколения.
- Высокая производительность при высоком числе организмов на одной карте.

Проектирование системы

Для того чтобы реализовать данный проект, было принято решение разделить его на две части: система моделирования и система принятия решений — ИНС, а также есть система, которая позволяет взаимодействовать с пользователем.

Система моделирования

Системой моделирования в данном проекте является набор классов, которые позволяют проводить некоторые расчеты и визуализировать процесс эволюции.

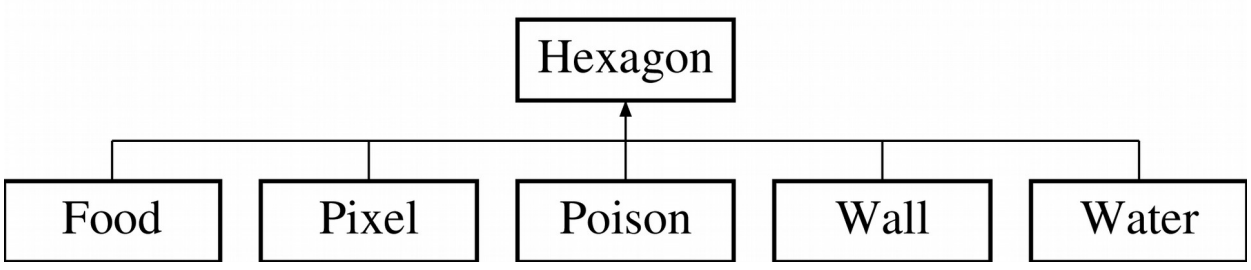
Основными классами в системе моделирования являются класс «Map» — класс самой карты, на которой все происходит, и класс «Hexagon» — класс шестиугольника, которые являются объектами карты.

Класс «Map»

Класс карты содержит в себе методы и объекты из библиотеки «SFML», которые позволяют визуально отражать процесс эволюции. Он также содержит в себе различные обработчики событий, которые происходят на карте: питание организмов, их гибель и тому подобное. Данный класс реализует генетический алгоритм в данном проекте: производит оценку, отбор и рекомбинацию организмов, а также запуск сохраненных поколений с произвольного номера — это позволяет не ждать определенного момента эволюции, а запустить произвольный этап. Карта состоит из указателей на шестиугольники, которые ее составляют.

Класс «Hexagon»

Класс объектов карты. Он является основным, так как от него наследуются все оставшиеся объекты карты.



- Класс «Pixel»

  Класс организма, пожалуй самый интересный объект на карте. Он имеет свой геном, который был описан выше, и подвергается действию процесса эволюции, он развивается (а точнее обучается и развивается искусственная нейронная сеть, которая заложена в каждом организме и движет всеми его действиями). Также организм питается, так его число жизней с каждым шагом уменьшается на 1, но он может употребить и яд, который приведет к болезни организма (проект был запущен на двух машинах, поэтому существует два вида яда: на одной машине эволюция была запущена с ядом, который приводит к болезни, которая ускоряет старение — жизни уменьшаются на 5 с каждым шагом, а на другой машине приводил сразу к гибели организма). После того как у организма заканчиваются жизни, он умирает. Если организм съел яд, значит он болел, следовательно, после гибели он становился ядом; если организм был здоров, то он становился едой. Отрисовывается на карте желтым цветом с цифрами посередине, которые показывают число жизней организма. Является подвижным объектом на карте.

- Класс «Food»

  Класс еды, которой питается организм. Основная функция — продление времени жизни организма. После того как организм съедает данный объект его число жизней увеличивается на 40. На карте отрисовывается зеленым цветом, является статическим объектом.

- Класс «Poison»

  Класс яда, который приводит к болезни организмов. На карте его количество равно количеству еды. На карте отрисовывается красным цветом, является статическим объектом.

- Класс «Wall»

  Класс стены. Они на карте созданы для того, чтобы организмы не вышли за пределы мира, в котором они живут. Эти объекты являются статическими, но не отрисовываются на карте.

- Класс «Water»

Класс воды, которая составляет основу карты. Изначально создается карта, состоящая из воды, по размеру, который задает пользователь. Эти объекты отрисовываются синим цветом и являются статическими.

Система принятия решений — ИНС

Систему принятия решений составляют классы, которые реализуют работу искусственной нейронной сети, которая позволяет организмам принимать решения. Основным классов здесь является класс «Brain».

Класс «Brain»

Этот класс является основным в системе принятия решений, так как объединяет в себе все остальные классы системы. Здесь происходит создание всех нейронов и связей в искусственной нейронной сети. Также он обеспечивает вызов методов классов, которые производят все вычисления, согласно расположению объектов карты вокруг организма. Класс «Brain» действительно является некоторым подобием мозга организма и обеспечивает принятие решения этого организма. В документации достаточно подробно расписаны все классы, входящие в систему принятия решений.

Класс «TrainAlgoritm»

Этот класс является не менее важным, чем класс «Brain», так как он обеспечивает тренировку — обучение нейронной сети. Обучение — это основной процесс данного проекта, так как моделирование эволюции здесь направлено на то, чтобы сеть научила организмов как можно дольше выживать в окружающем их мире, чтобы он обходили яд стороной и питались исключительно едой, которая не наносит им вреда.

Система взаимодействия

Систему взаимодействия составляют 3 класса: «Evolution», «Button» и «Keyboard». Они позволяют упростить работу пользователя, запускающего моделирование.

Класс «Evolution»

Данный класс соединяет в себе систему моделирования и систему принятия решений, а также запускает отрисовку всех объектов. В этом классе также есть метод, который создает меню, позволяющее менять размеры карты, запускать новую эволюцию или запустить ее с произвольного поколения.

Класс «Button»

Класс описывает кнопки, которые расположены на том же окне где отрисована карта. Они позволяют ускорять или замедлять движение организмов на карте. Это позволяет при замедлении подробно рассмотреть движение организмов на карте.

Класс «Keyboard»

Класс обеспечивающий взаимодействие с клавиатурой, так как она позволяет управлять кнопками и меню.

Выбор технологий

Выбор языка программирования

В разработке любого проекта очень важной частью является выбор языка программирования. Этот выбор заключается в том, что нужно подобрать такой язык, который будет оптимальным для конкретной разработки. Если планируется разработка глобального проекта, то он будет состоять и достаточно большого количества подпроектов, для каждого из которых будет выбран язык, который наилучшим образом будет справляться с поставленными задачами.

Важными критериями при выборе являются:

- Размер и тип проекта
- Сложность проекта
- Скорость разработки
- Доступные инструменты разработки
- Наличие готовых решений
- Гибкость решения
- Наличие подробной документации
- Требования к нагрузкам
- Требования к безопасности
- Кроссплатформенность
- Возможность интеграции с другими решениями

Данный проект не планировался очень объемным с ограниченным временем на разработку, поэтому выбирался лишь один язык программирования.

Всем известно, что языки программирования разделяются по сфере применения. Основными сферами являются веб-разработка, мобильная и игровая разработка. Самыми популярными языками в веб-разработке на данный момент являются: HTML, CSS, JavaScript, Java, Python и PHP . В разработке мобильных приложений на Android: Java, на iOS: Swift и Objective-C. И наконец в разработке игр: C#, C++, JavaScript, Java, Smalltalk. Данный проект относиться к сфере разработке игр, поэтому выбор был из

последнего списка ЯП, технологии которого относятся к объектно-ориентированным языкам программирования.

После рассмотрения языков программирования, было принято решение использовать язык C++.

Рассмотрим выбранный язык программирования C++.

Следует пожалуй начать с достоинств этого языка:

C++ — чрезвычайно мощный язык, содержащий средства создания эффективных программ практически любого назначения, от низкоуровневых утилит и драйверов до сложных программных комплексов самого различного назначения. В частности:

- Высокая совместимость с языком C, позволяющая использовать весь существующий C-код (код C может быть с минимальными переделками скомпилирован компилятором C++; библиотеки, написанные на C, обычно могут быть вызваны из C++ непосредственно без каких-либо дополнительных затрат, в том числе и на уровне функций обратного вызова, позволяя библиотекам, написанным на C, вызывать код, написанный на C++).
- Поддерживаются различные стили и технологии программирования, включая традиционное директивное программирование, ООП, обобщенное программирование, метапрограммирование (шаблоны, макросы).
- Имеется возможность работы на низком уровне с памятью, адресами, портами.
- Возможность создания обобщённых контейнеров и алгоритмов для разных типов данных, их специализация и вычисления на этапе компиляции, используя шаблоны.
- Кроссплатформенность. Доступны компиляторы для большого количества платформ, на языке C++ разрабатывают программы для самых различных платформ и систем.

- Эффективность. Язык спроектирован так, чтобы дать программисту максимальный контроль над всеми аспектами структуры и порядка исполнения программы.

Конечно же у всех языков имеются недостатки, C++ не исключение:

Отчасти недостатки C++ унаследованы от языка-предка — Си — и вызваны изначально заданным требованием возможно большей совместимости с Си. Это такие недостатки, как:

- Синтаксис, провоцирующий ошибки:
- Препроцессор, унаследованный от C, очень примитивен.
- Плохая поддержка модульности (по сути, в классическом Си модульность на уровне языка отсутствует, её обеспечение переложено на компоновщик). Подключение интерфейса внешнего модуля через препроцессорную вставку заголовочного файла (#include) серьёзно замедляет компиляцию при подключении большого количества модулей (потому что результирующий файл, который обрабатывается компилятором, оказывается очень велик).

Выбор используемых библиотека

Существует много графических библиотек на C++. Основными являются OpenGL, QT, SFML и SDL.

При разработке данного проекта требовалась простоя и быстрая библиотека, так как графика не является основной в работе, а нужна лишь для визуализации. Также графика не должна замедлять работу программы.

Библиотека QT не подходит данным требованиям, так как у нее достаточно медленная работа с графикой.

Библиотека SDL не поддерживает ООП, поэтому даже не рассматривалась при выборе библиотеки.

Библиотека OpenGL является абсолютно графической библиотекой, у нее нет поддержки открытия окон, взаимодействия с клавиатурой, что тоже является большим минусовм.

Библиотека SFML является самой оптимальной библиотекой для данного проекта.

SFML очень простая и понятная библиотека, которая испольеутся при разработке игр. Она обеспечивает простой интерфейс для различных компонентов ПК и также поддерживает ООП. Библиотека имеет официальную привязку к языкам C и имеет понятную документацию с примерами.

Описание технических решений

Во время разработки проекта возникло несколько проблем.

Первая проблема связана с тем, что была ошибка клонирования организмов и на большом числе поколений на карте находилось очень много организмов и в связи с этим работа программы тормозилась. Ошибка быстро и просто была устранена.

Вторая проблема была связана со скоростью смены поколений. Смена происходила достаточно медленно, но проблема также решилась быстро. Было принято решение отключать отрисовку если скорость у организмов меньше выбранного порогового значения.

Третья проблема была связана со строением нейронной сети и последующем ее обучении. Приходилось пробовать создавать различное количество нейронов во входном и выходном слоях. Поэтому экспериментальным путем пришли к выводу, что самое оптимальное количество нейроном во входном и выходном слоях 10 и 7 нейронов соответственно.

Заключение

В итоге, несмотря на возникшие проблемы, удалось реализовать достаточно успешный проект, который выполняет необходимые функции: позволяет увидеть работу генетических алгоритмов, позволяет организмов из произвольных поколений запускать на произвольные карты, позволяет свободно менять размеры карты, на которой будет происходить эволюция.

В дальнейшем планируется улучшить систему реализации, провести эксперименты с различными функциями активации нейронов и проанализировать скорость обучения нейронной сети.

Список используемых источников

1. Документация: Стандартные библиотеки C++ [Электронный курс]. URL: www.cplusplus.com/reference/
2. Документация: Официальная документация по языку программирования C++ [Электронный курс]. URL: http://ru.cppreference.com/w/
3. Документация: Официальная документация по фреймворку SFML [Электронный курс]. URL: https://www.sfml-dev.org/
4. М. Тим Джонс «Программирование искусственного интеллекта в приложениях» Второе издание 2011 г.
5. Laurene Fausett «Fundamentals of Neural Networks»