ИСКУССТВЕННАЯ ЖИЗНЬ С ИСПОЛЬЗОВАНИЕМ ГЕНЕТИЧЕСКОГО АЛГОРИТМА

Д. И. Исайкин

Учреждение образования «Гомельский государственный технический университет имени П. О. Сухого», Республика Беларусь

Научный руководитель Г. П. Косинов

Целью исследования является применение генетического алгоритма для создания бота, который способен прожить миллион итераций в мире с заданными правилами.

Итак, генетический алгоритм – это, прежде всего, метод многомерной оптимизации, т. е. метод поиска минимума многомерной функции. Потенциально этот метод можно использовать для глобальной оптимизации, но с этим возникают сложности, опишу их позднее.

Сама суть метода заключается в модулировании эволюционного процесса: существует какая-то популяция (набор векторов), размножающаяся, на которую воздействуют мутации, и производится естественный отбор на основании минимизации целевой функции. Рассмотрим подробнее эти процессы.

Прежде всего популяция должна **размножаться**. Основной принцип размножения – потомок похож на своих родителей, т. е. необходимо задать какой-то механизм наследования, и лучше будет, если он будет включать элемент случайности. Но скорость развития таких систем очень низкая – разнообразие генетическое падает, популяция вырождается, т. е. значение функции перестает минимизироваться.

Для решения этой проблемы был введен механизм **мутации**, который заключается в случайном изменении каких-то особей. Этот механизм позволяет привнести что-то новое в генетическое разнообразие.

Следующий важный механизм – **селекция**. Как было сказано, селекция – отбор особей, которые лучше минимизируют функцию. Обычно отбирают столько особей, сколько было до размножения, чтобы из эпохи в эпоху у нас было постоянное количество особей в популяции. Также иногда отбирают «счастливчиков» – некоторое число особей, которые хоть и плохо минимизируют функцию, но зато могут внести разнообразие в последующие поколения.

Этих трех механизмов обычно недостаточно, чтобы минимизировать функцию. Так популяция вырождается – рано или поздно локальный минимум забивает своим значением всю популяцию. Когда такое происходит, проводят процесс, называемый **встряской** (в природе аналогии этому – глобальные катаклизмы), когда уничтожается почти вся популяция, и добавляются новые (случайные) особи.

В качестве инструмента в реализации алгоритма был выбран язык программирования высокого уровня – javascript, среда разработки – Visual Studio Code.

Существует мир со своими правилами, главные создания здесь – боты, у них есть здоровье, и каждый ход оно уменьшается на 1. Чтобы выжить, боту необходимо найти еду, которая прибавляет 10 к здоровью. Если оно опустилось до нуля – бот умирает. Еду можно съесть, зайдя на клетку с едой, или схватить ее из соседней клетки. Так же есть стены и яд, если бот зайдет на клетку с ядом, то он погибнет, но есть возможность преобразовать яд в еду. Необходимо найти такого бота, который сможет прожить 1.000.000 шагов.

При первом запуске случайным образом создается 64 бота, после чего они начинают жить в этом мире. Когда останется 8 штук – создастся новое поколение из оставшихся, каждый выживший даст 8 полностью идентичных потомков, 1 из которых случайно мутирует, поменяется одна или несколько клеток ДНК случайным образом, и все пойдет по новому кругу.

ДНК бота – это 64 ячейки, замкнутые по кругу. При первом запуске эти ячейки заполняются случайными числами от 0 до 63. Каждое число – это определенная команда, которая выполняет свою операцию:

1) 0–7 – перемещение бота на соседнюю клетку; направление зависит от числа и от того, куда смотрит бот, указатель текущей команды в коде перейдет количество клеток, зависящее от того, что было по этому направлению (яд – 1 клетка; стена – 2; бот – 3; еда – 4; пусто – 5). Данная команда является завершающей и после нее управление перейдет к следующему боту;

2) 8–15 – схватить; если в указанном направлении была еда – он ее съест, если яд – преобразовывает в еду, в остальных случаях ничего не происходит, указатель команды перемещается так же как и в предыдущем пункте. Данная команда тоже является завершающей;

3) 16–23 – посмотреть; бот остается на месте, а указатель перемещается в зависимости от увиденного;

4) 24–31 – поворот; указатель переходит на следующую ячейку;

5) 32–63 – безусловный переход; указатель перепрыгивает на столько ячеек вперед, какое число представляет команда.

Команды 3–5 не заканчивают ход, однако за ход может быть выполнено не более 10 команд, после чего управление передастся следующему боту.

Также команды 1–3 зависят от того, куда был повернут бот.

После запуска мира первые поколения (в основном боты) либо стоят на месте, либо передвигаются по двум клеткам. Некоторое время стояние на месте будет оставаться выгодной стратегией, так как меньше шансов наткнуться на яд. Это будет оставаться неизменным до тех пор, пока боты не научатся преобразовывать яд в еду, тогда станет выгодней постоянно перемещаться.

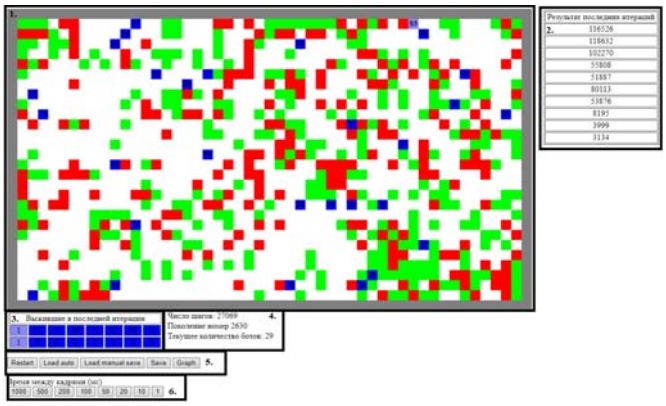

Рис. 1. Внешний вид приложения

Теперь кратко опишем дизайн (рис. 1). Здесь пользователь видит следующее:

1) поле с ботами;

2) таблицу, в которой показан результат 10 последних итераций;

3) таблицу с выжившими в последней итерации (их здоровье и поколений без мутаций);

4) число шагов, номер поколения и количество ботов;

5) также есть кнопки управления:

– Restart – пересоздает ботов со случайным ДНК;

– Load auto – загружает последнее автосохранение (программа сохраняет ботов каждое новое поколение);

– Load manual save – загружает последнее ручное сохранение;

– Save – сохраняет ботов после селекции;

– Graph – выводит график всех итераций (рис 2);

6) кнопки для управления длительности времени между каждым шагом.

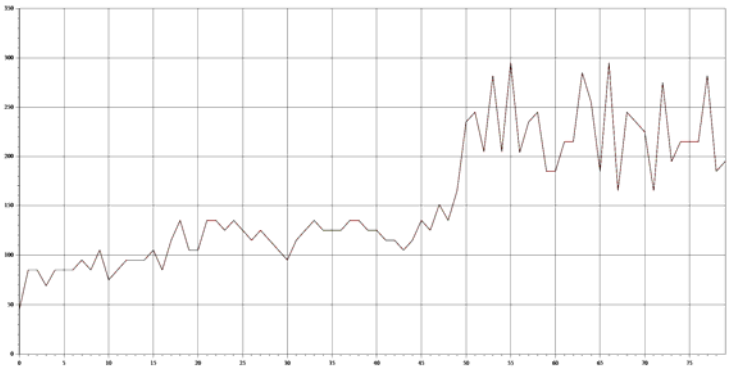

Рис. 2. График всех итераций

Таким образом, в данной среде боты спокойно живут и развиваются. К сожалению, скорость работы программы – небольшая, примерно 125 шагов в секунду, поэтому на данный момент рекордное значение равно 150.000 шагов. Программа со своей задачей справляется, а значит, остается просто ждать, пока они смогут развиться до необходимого уровня.