N лучших различных особей среди родителей и детей. При этом дублирование генотипов не допускается. В модели CHC размер популяции относительно мал — около 50 особей. Это оправдывает использование однородного кроссинговера и позволяет алгоритму сойтись к решению. CHC противопоставляет агрессивный отбор агрессивному кроссинговеру, однако, малый размер популяции быстро приводит ее к состоянию, когда создаются только более или менее одинаковые генотипы. В таком случае CHC применяет cataclysmic mutation: все генотипы, кроме самого приспособленного, подвергаются сильной мутации (изменяется около трети битов). Так алгоритм перезапускается и далее продолжает работу, применяя только кроссинговер. Таким образом, алгоритм CHC довольно быстро сходится из-за того, что в нем нет мутаций, следующих за оператором кроссинговера, используются популяции небольшого размера, и отбор особей в следующее поколение ведется и между родительскими особями, и между их потомками.

Идея гибридных алгоритмов заключается в сочетании генетического алгоритма с некоторым другим классическим методом поиска, подходящим в данной задаче. В каждом поколении все сгенерированные потомки оптимизируются выбранным методом и затем заносятся в новую популяцию. Тем самым получается, что каждая особь в популяции достигает локального оптимума, вблизи которого она находится. Далее, производятся обычные для ГА действия: отбор родительских пар, кроссинговер и мутации. На практике гибридные алгоритмы оказываются очень удачными. Это связано с тем, что вероятность попадания одной из особей в область глобального максимума обычно велика. После оптимизации такая особь будет являться решением задачи. Известно, что генетический алгоритм способен быстро найти во всей области поиска хорошие решения, но он может испытывать трудности в получении из них наилучших. Обычный оптимизационный метод может быстро достичь локального максимума, но не может найти глобальный. Сочетание двух алгоритмов позволяет использовать преимущества обоих.

В модели генитор используется специфичный способ отбора. Вначале, как и полагается, популяция инициализируется, и ее особи оцениваются. Затем выбираются случайным образом две особи, скрещиваются, причем, получается только один потомок, который оценивается и занимает место менее приспособленной особи в популяции, а не одного из родителей. После этого снова случайным образом выбираются две особи, и их потомок занимает место родительской особи с самой низкой приспособленностью. Таким образом, на каждом шаге в популяции обновляется лишь одна особь. Процесс продолжается до тех пор, пока пригодности хромосом не станут одинаковыми. В данный алгоритм можно добавить мутацию потомка после его создания. Критерий окончания процесса, как и вид кроссинговера и мутации, можно выбирать разными способами.

Для повышения эффективности работы ГА создано множество модификаций. Они связаны с применением иных методов селекции, с модификацией генетических операторов, с преобразованием функции приспособленности, а также, с различными способами кодирования параметров задачи в форме хромосом. Описанные выше примеры не исчерпывают многообразие областей и задач применения ГА. Эта область сейчас является перспективной и требует ещё долгого изучения.

Список использованных источников:

1. Эволюционные вычисления [Электронный ресурс]. – Режим доступа к источнику: https://www.intuit.ru/studies/courses/14227/1284/info

2. Гладков Л.А., Курейчик В.В., Курейчик В.М. Генетические алгоритмы // Под ред. В.М. Курейчика. — 2-е изд., испр. и доп. — Москва: ФИЗМАТЛИТ, 2006. — 320 с.

РЕАЛИЗАЦИЯ ГОЛОВОЛОМКИ «КУБИК РУБИКА»

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Камеко В.В.

Данилова Г. В. – м. т. н., ассистент

В настоящий момент, в связи со спросом на головоломки различных видов, возникла потребность в их симуляторах, позволяющих собирать головоломки на компьютерах. В качестве головоломки для реализации была выбрана самая известная из них – «Кубик Рубика». В программе будет реализованы все функции данной головоломки на языке программирования Delphi.

Известный факт, что головоломки положительно влияют на развитие пространственного и логического мышления, а также, являются отличным антистрессом. Одна из таких головоломок, «Ку́бик Ру́бика», не только стала самой знаменитой головоломкой и собрала целое движение «спидкуберов», но и поставила довольно интересную задачу математикам и программистам всего мира. В настоящее время существует достаточно мало приложений эмулирующих эту замечательную игрушку, как и не существует единого формата хранения состояния «Кубика».

«Ку́бик Ру́бика» – механическая головоломка, изобретённая в 1974, представляющая из себя куб 3x3x3 с окрашенными в разные цвета гранями. Задача игрока – собрать «кубик» в первоначальное состояние из перемешанного.

Рис. 1 – Различные вариации головоломки Кубика Рубика.

Существуют также различные вариации данной головоломки, например: головоломки с произвольными размерами (NxMxK), кубики других многогранников (додекаэдр, тетраэдр, усеченные пирамиды, призмы, октаэдр, ромбододекаэдр, кубооктаэдр), а также многомерные аналоги.

В программе будут реализованы следующие функции:

- Возможность сохранения и загрузки состояний кубика.
- Поддержка нескольких сохранений.
- Возможность изменения состояния кубика, путем вращения граней.
- Поворот самого кубика для лучшего представления о его положении в пространстве.

Для реализации сохранений будет спроектирован специальный формат файла, поддерживающий хранение состояния кубика, но не всю историю его изменения, для экономии размера файла и упрощения взаимодействия пользователя с программой.

Для полного погружения в игру планируется добавить очки и время, потраченное на сборку кубика.

Планируется добавить фоновую музыку и главное меню, где пользователь сможет выбирать способы взаимодействия с программой. В планируемые функции также входит добавление возможности выбора размера кубика и цвета его граней.

Управление будет реализовано через кнопки интерфейса и движений мышью для поворота кубика или его граней. Интерфейс и другие изображения по возможности будут нарисованы.

Программу планируется создавать на игровом движке с открытым исходным кодом «Castle Engine», для облегчения работы с созданием интерфейса пользователя и 3D-объектами. В программе будут использованы кнопки для взаимодействия с пользователем, различного вида картинки, а также 3D-модель кубика. Исходный код, по возможности, будет разбит на модули со схожей функциональностью.

Список использованных источников:

1. https://ru.wikipedia.org/wiki/Кубик_Рубика – Кубик Рубика. Материал из Википедии
2. https://ru.wikipedia.org/wiki/Пирамидка_Мефферта – Пирамидка_Мефферта. Материал из Википедии
3. https://castle-engine.io – Бесплатный open-source 3D и 2D игровой движок

АЛГОРИТМЫ И СИСТЕМЫ РАСЧЕТА БУКМЕКЕРСКИХ КОЭФФИЦИЕНТОВ НА СПОРТИВНЫЕ СОБЫТИЯ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Канунников И.С.
Парамонов А.И. – канд. техн. наук, доцент

Для эффективной работы букмекерских контор существует такая структура в их организации, как аналитический отдел, главная задача которого – рассчитать и грамотно расставить базовые коэффициенты на события. Аналитик, применяя математический аппарат теории вероятности, статистического анализа и численных методов, выставляет коэффициент возможности того или иного события. Безошибочно работающий аналитический отдел – непременный залог прибыльности букмекерской конторы.

На начальном этапе деятельность аналитического отдела связана со сбором информации, такой как статистические показатели игроков, команд и прочее. Затем выполняется её обработка и формирование спортивной линии. Обычно, за несколько дней до события, формируются начальные коэффициенты, которые затем корректируются таким образом, чтобы ставки были разделены в пропорции 50 на 50 (для упрощенного варианта события с двумя возможными исходами – например, победа или поражение команды). Предположим, требуется вычислить первичный коэффициент на домашнюю победу футбольного клуба А над клубом Б. Формирование коэффициентов упрощенно проходит по следующему алгоритму: пусть Клуб А