**БУДУТ ВОПРОСЫ МОЖЕШЬ СЕГОДНЯ ДО 22 ЧАСОВ ЗВОНИТЬ**

**Введение**

Для решения оптимизационных задач в последнее время всё чаще применяются генетические алгоритмы. Генетический алгоритм – это эвристический алгоритм поиска, используемый для решения задач оптимизации и моделирования путём случайного подбора и комбинирования искомых параметров, основанный на принципах естественного отбора Ч. Дарвина. Генетические алгоритмы были предложены Джоном Холландом в 1975 году .Они полезны при решении таких задач, как: оптимизация функций и запросов в базах данных, разнообразные задачи на графах (задача коммивояжера, раскраска, нахождение паросочетаний и т. д.), составление расписаний, игровые стратегии, а также в искусственной нейронных сетях.

**Цель работы**: создать универсальную и несложную во внедрении библиотеку, реализующую генетические алгоритмы.

Для достижения данной цели нами были поставлены и решены следующие **задачи**:

Изучить основные принципы и области применения генетических алгоритмов.

Разработать библиотеку, позволяющую легко интегрировать функции генетических алгоритмов в любое приложение. При этом сделать структуру библиотеки максимально универсальной и простой в использовании.

Протестировать работу библиотеки на примере нескольких приложений и оценить эффективность использования генетических алгоритмов.

**Глава 1. Основные понятия и принципы работы генетических алгоритмов**

1. Основные понятия *(Выкинуть не используемое после написания остального текста)*

Введем основные понятия, применяемые в генетических алгоритмах.

Вектор — упорядоченный набор чисел, называемых компонентами

вектора

3

. Так как вектор можно представить в виде строки его координат,

то в дальнейшем понятия вектора и строки считаются идентичными.

Булев вектор — вектор, компоненты которого принимают значения из

двух элементного (булева) множества, например, {0, 1} или {−1, 1}.

Хеммингово расстояние — используется для булевых векторов и равно

числу различающихся в обоих векторах компонент.

Хеммингово пространство — пространство булевых векторов, с введен-ным на нем расстоянием (метрикой) Хемминга. В случае булевых векторов

размерности n рассматриваемое пространство представляет собой множе-ство вершин n-мерного гиперкуба с хемминговой метрикой. Расстояние

между двумя вершинами определяется длиной кратчайшего соединяющего

их пути, измеренной вдоль ребер.

Хромосома — вектор (или строка) из каких-либо чисел. Если этот век-тор представлен бинарной строкой из нулей и единиц, например, 1010011,

то он получен либо с использованием двоичного кодирования, либо кода

Грея (см. Приложение A.3). Каждая позиция (бит) хромосомы называется

геном.

Индивидуум (генетический код, особь) — набор хромосом (вариант ре-шения задачи). Обычно особь состоит из одной хромосомы, поэтому в даль-нейшем особь и хромосома идентичные понятия.

Расстояние — хеммингово расстояние между бинарными хромосомами.

Кр ´ оссинговер (кроссовер) — операция, при которой две хромосомы об-мениваются своими частями. Например, 1100&1010 −→ 1110&1000.

Мутация — случайное изменение одной или нескольких позиций в хро-мосоме. Например, 1010011 −→ 1010001.

Инверсия — изменение порядка следования битов в хромосоме или в ее

фрагменте. Например, 1100 −→ 0011.

Популяция — совокупность индивидуумов.

Пригодность (приспособленность) — критерий или функция, экстремум

которой следует найти.

Локус — позиция гена в хромосоме

Аллель — совокупность подряд идущих генов.

Эпистаз — влияние гена на пригодность индивидуума в зависимости

от значения гена, присутствующего в другом месте. Ген считают эписта-3

Заметим, что данное определение вектора отличается от определения, принятого в мате-матике.

12

тическим, когда его присутствие подавляет влияние гена в другом локусе.

Эпистатические гены из-за их влияния на другие гены иногда называют

ингибирующими. Подавление проявления гена неаллельным ему геном на-зывается гипостазом, а сам подавляемый ген — гипостатическим.

Из приведенных выше определений следует, что терминология ГА пред-ставляет собой синтез собственно генетических и искусственных понятий.

Так, для понятия, заимствованного из генетики, можно предъявить его ис-кусственный (символический) аналог. Например, хромосома и строка. В

биологических системах полный генетический пакет

4

называется геноти-пом. В искусственных системах полный генетический пакет строк называ-ется структурой. В биологических системах в процессе индивидуального

развития организма взаимодействие генотипа с окружающей средой форми-рует совокупность внешних признаков и свойств, называемую фенотипом.

В математическом моделировании рассматриваемая структура декодируется

с помощью множества параметров, которое в литературе иногда называ-ют альтернативным решением или точкой. Всевозможные значения пара-метров образуют пространство решений. В искусственной генетической

системе возможно использование как числовых, так и нечисловых пара-метров.

В биологической терминологии говорят, что хромосома образована гена-ми. В генетике с любым локусом связана определенная генетическая функ-ция. Поэтому можно говорить о специализированных генах. Например, ген

цвета глаз животного находится в 10 локусах, т.е. голубой цвет глаз имеет

10-аллельное значение. В терминологии ГА говорят, что строки образованы

значениями функции, или детекторами. Значения функции могут быть

локализованы в различных позициях строки. Связь между естественной

(биологической) и искусственной терминологией приведена в таблице

Основные принципы работы генетических алгоритмов можно отобразить в следующей схеме:

Генерирование начальной популяции

Вычисление пригодности индивидов

Селекция

Мутация

Скрещивание

Достигнут требуемый результат?

**Глава 2. Библиотека**

Три основных класса - Generation (Поколение), Individual (Индивид) и Dna (ДНК). В них реализован основные операции генетических алгоритмов (селекция, отбор, мутация), а также хранятся параметры для алгоритма (тип селекции, вероятность мутации и т.д.). Для внедрения генетических алгоритмов разработчику необходимо лишь создать переменную типа Generation, указав в качестве типа Individual. При этом тип индивида необходимо выбрать в зависимости от конкретной задачи из числа уже реализованных в библиотеке. Если же ни один из реализованных типов не походит к поставленной задаче, разработчик может создать свой класс, базирующийся на Individual из нашей библиотеки. Для этого класса ему необходимо определить только функцию подсчёта приспособленности и тип используемой ДНК, остальное наследуется от базового класса и не требует переопределения. Разуметься, если ДНК также необходимо специфическое, его можно реализовать самому, определив для него методы мутации и генерации. Перед использованием необходимо также задать настройки генетического алгоритма. Стоит отметить, что в ходе работы программы их можно изменять (например, увеличивать с каждым новым поколением вероятность мутации). Такой подход к работе с библиотекой даёт разработчику простор для экспериментов и позволяет найти наиболее оптимальные настройки для алгоритма в поставленной задаче. Как и положено, к библиотеке прилагается документация в форматах LaTeX, HTML и PDF. Это позволит разработчику быстро понять её структуру и внедрить её в свой проект.

На данный момент в библиотеке кроме базовых классов также реализованы следующие типы ДНК:

* Линейная действительная
* Линейная действительная с ограниченным диапазоном
* Линейная бинарная
* Древовидная ДНК выражения

В ней также можно найти индивидов для решения таких задач, как:

* Нахождение минимума / максимума функций
* Подбор функций под заданные значения переменных и заданный результат
* Подбор параметров машины для прохождения заданной трассы.

*Тут описать все типы поддерживаемого скрещивания, выбора родителей и селекции*

Структурная схема библиотеки выглядит следующим образом:

*- Тут будет UML схема -*

**Глава 3. Примеры использования**

1. Минимум/максимум функции

Данный пример является простейшей иллюстрацией работы генетического алгоритма. Пользователь задаёт какую-либо функцию, а задачей компьютера является нахождение её минимум/максимума. Стоит отметить, что для сложных функций результат в данном примере в большой степени зависит от мутаций, так как они помогают алгоритму находить минимум/максимум на всей области определения, а не останавливаться на локальных экстремумах. Результаты работы на различных функция приведены в таблице ниже:

*--- Таблицы с результатами для разных функций (погрешность и попдание в локальный экстремум для разных запусков + среднее)---*

1. Тест\_индивид
2. Подбор функции

В данной задаче в качестве постоянных значений известны значения переменных и результат выражения. Программе необходимо подобрать выражение, при подстановке в которое значений переменных результат будет как можно ближе к заданному. Анализируя точность работы программы, мы можем составить следующую таблицу:

*--Тут исходная функция и значения переменных ---*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Исходная функция | Переменные | № запуска | № поколения | Лучший индивид | Его значение | Погрешность |
|  |  |  |  |  |  |  |
|  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |  |  |

1. Машинки

Эта задача, пожалуй, самая зрелищная из всех примеров. Постоянной величиной здесь является случайно сгенерированная трасса, представленная ломаной. В качестве индивида здесь выступает машина. Машина представляет собой набор векторов из одной очки, на которые «натянут» многоугольник, называемый «корпусом». В некоторых вершинах этого многоугольника находятся колёса, которые соединены с вершиной с помощью пружины. ДНК машины хранит длину каждого из векторов, радиус каждого из колёс, его наличие либо отсутствие в конкретной вершине, а также жёсткость соединительной пружины. При этом для каждого из параметров пользователь задаёт минимальное и максимальное значение. Примеры машин из разных поколений вы можете увидеть на рисунках, расположенных ниже:

*-=Тут скриншоты с разных поколений =-*

**Заключение**

В результате нашей работы, нами были:

* + Изучены генетические алгоритмы
  + Разработана библиотека
  + Библиотека была успешно интегрирована в ряде приложений
  + Была изучена эффективность использования ГА

В дальнейшем мы планируем:

* + Добавить новые компоненты в библиотеку
  + Оптимизировать библиотеку

**Список литературы**