Генетические алгоритмы

Генетические алгоритмы

Понятие генетического алгоритма

Генети́ческий алгори́тм (англ. genetic algorithm) — это эвристический ***алгоритм поиска***, применяемый для решения задач оптимизации и моделирования ***путем последовательного подбора, комбинирования и вариации*** искомых параметров с использованием механизмов, напоминающих биологическую эволюцию. Идея генетических алгоритмов заимствована у живой природы и состоит в организации эволюционного процесса, конечной целью которого является получение оптимального решения в сложной комбинаторной задаче. Разработчик генетических алгоритмов выступает в данном случае как "создатель", который должен правильно установить законы эволюции, чтобы достичь желаемой цели как можно быстрее. В генетическом алгоритме используются как аналог механизма генетического наследования, так и аналог естественного отбора. При этом сохраняется биологическая терминология в упрощенном виде.

Принцип работы ГА

Задача кодируется таким образом, чтобы её решение могло быть представлено в виде вектора («хромосома»). Случайным образом создаётся некоторое количество начальных векторов («начальная популяция»). Они оцениваются с использованием «функции приспособленности», в результате чего каждому вектору присваивается определённое значение («приспособленность»), которое определяет вероятность выживания организма, представленного данным вектором. После этого с использованием полученных значений приспособленности выбираются вектора (селекция), допущенные к «скрещиванию». К этим векторам применяются «генетические операторы» (в большинстве случаев «скрещивание» - crossover и «мутация» - mutation), создавая таким образом следующее «поколение». Особи следующего поколения также оцениваются, затем производится селекция, применяются генетические операторы и т. д. Так моделируется «эволюционный процесс», продолжающийся несколько жизненных циклов (поколений), пока не будет выполнен критерий остановки алгоритма.

Таким критерием может быть:

- нахождение глобального, либо субоптимального решения;
- исчерпание числа поколений, отпущенных на эволюцию;
- исчерпание времени, отпущенного на эволюцию.

Этапы генетического алгоритма

- Создание начальной популяции
- Вычисление функций приспособленности для особей популяции (оценивание)

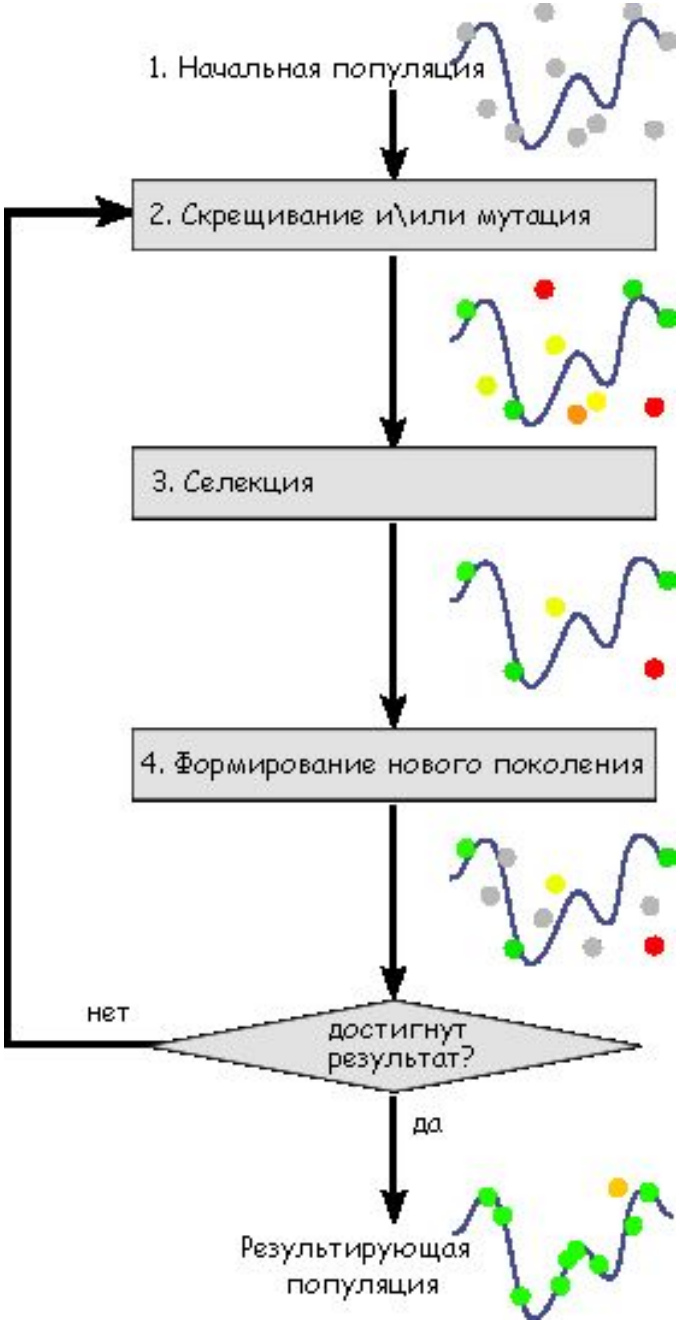
Начало цикла:

- Выбор индивидов из текущей популяции (селекция)
- Скрещивание и\или мутация
- Вычисление функций приспособленности для всех особей
- Формирование нового поколения
- Если выполняются поставленные критерии, то

Конец цикла

иначе Начало цикла.

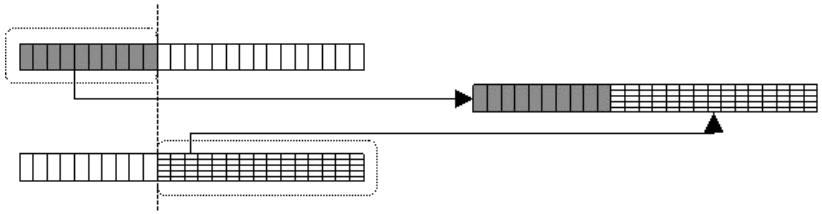
Модель «эволюционного процесса»



Простой генетический алгоритм

Основные операции генетических алгоритмов

Операция скрещивания. Скрещивание является главной генетической операцией. Эта операция выполняется над двумя хромосомами- родителями и создает отпрыск путем комбинирования особенностей обоих родителей. Приведем простейший пример скрещивания. В начале выберем некоторую случайную точку (точка скрещивания - англ. cut-point), после этого создадим хромосому-отпрыск путем комбинирования сегмента первого родителя, стоящего слева от выбранной точки скрещивания, с сегментом второго родителя, стоящего по правую сторону от точки скрещивания, как это показано на рис



Доля производимых на каждой итерации отпрысков называется **коэффициентом скрещивания**. Произведение коэффициента скрещивания на размер популяции показывает количество отпрысков. Большое значение этого коэффициента позволяет исследовать больше областей пространства поиска (или пространства решений) и уменьшает шанс попадания в локальный минимум. Но если значение слишком велико, то это приведет к большим затратам времени вычислений на исследование бесперспективных областей.

Операция мутации. Мутация - это фоновая операция, производящая случайное изменение в различных хромосомах. Наипростейший вариант мутации состоит в случайном изменении одного или более генов. В ГА мутация играет важную роль для:

а) восстановления генов, выпавших из популяции в ходе операции выбора, так что они могут быть опробованы в новых комбинациях,
б) формирования генов, которые не были представлены в исходной популяции.

Интенсивность мутаций определяется коэффициентом мутаций. Он представляет собой долю генов, подвергающихся мутации на данной итерации, в расчете на их общее число. Слишком малое значение этого коэффициента приводит к тому, что многие гены, которые могли бы быть полезными, никогда не будут рассмотрены. В то же время слишком большое значение коэффициента приведет к большим случайным возмущениям. Отпрыски перестанут быть похожими на родителей и алгоритм потеряет возможность обучаться, сохраняя наследственные признаки .

Стратегии поиска. Поиск является одним из наиболее универсальных методов нахождения решения для случаев, когда априори не известна последовательность шагов, ведущая к оптимуму. Существуют две поисковые стратегии: эксплуатация наилучшего решения и исследование пространства решений. Градиентный метод является примером стратегии, которая выбирает наилучшее решение для возможного улучшения, игнорируя в то же время исследование всего пространства поиска.

Случайный поиск является примером стратегии, которая, наоборот, исследует пространство решений, игнорируя исследование перспективных областей поискового пространства.

Генетический алгоритм представляет собой класс поисковых методов общего назначения, которые комбинируют элементы обоих стратегий. Использование этих методов позволяет удерживать приемлемый баланс между исследованием и эксплуатацией наилучшего решения. В начале работы генетического алгоритма популяция случайна и имеет разнообразные элементы. Поэтому оператор скрещивания осуществляет обширное исследование пространства решений. С ростом значения функции соответствия получаемых решений оператор скрещивания обеспечивает исследование окрестностей каждого из них. Другими словами, тип поисковой стратегии (эксплуатация наилучшего решения или исследование области решений) для оператора скрещивания определяется разнообразием популяции, а не самим этим оператором.

Преимущества генетических алгоритмов

Существуют два главных преимущества генетических алгоритмов перед классическими оптимизационными методиками:

1. **ГА не имеет значительных математических требований к видам целевых функций и ограничений.** Исследователь не должен упрощать модель объекта, теряя ее адекватность, и искусственно добиваясь возможности применения доступных математических методов. При этом могут использоваться самые разнообразные целевые функции и виды ограничений (линейные и нелинейные), определенные на дискретных, непрерывных и смешанных универсальных множествах.
2. При использовании классических пошаговых методик глобальный оптимум может быть найден только в том случае когда проблема обладает свойством выпуклости. В тоже время **эволюционные операции генетических алгоритмов позволяют эффективно отыскивать глобальный оптимум.**

Пример ГА: Решение Диофантова уравнения

Рассмотрим диофантово (только целочисленные решения) уравнение:
a+2b+3c+4d=30, где a, b, c и d – некоторые положительные целые.
Применение ГА за очень короткое время находит искомое решение (a, b, c, d).
Для начала выберем 5 случайных решений: 1 =< a,b,c,d =< 30

Таблица 1: 1-е поколение хромосом и их содержимое

Хромосома	**(a,b,c,d)**
1	(1,28,15,3)
2	(14,9,2,4)
3	(13,5,7,3)
4	(23,8,16,19)
5	(9,13,5,2)

Чтобы вычислить коэффициенты выживаемости (fitness), подставим каждое решение в выражение *a+2b+3c+4d*. Расстояние от полученного значения до 30 и будет нужным значением.

Таблица 2: Коэффициенты выживаемости первого поколения хромосом

Хромосома	Коэффициент выживаемости
1	|114-30|=84
2	|54-30|=24
3	|56-30|=26
4	|163-30|=133
5	|58-30|=28

Так как меньшие значения ближе к 30, то они более желательны (приспособленность).

В нашем случае большие численные значения коэффициентов выживаемости подходят, увы, меньше.

Чтобы создать систему, где хромосомы с более подходящими значениями имеют большие шансы оказаться родителями, мы должны вычислить, с какой вероятностью (в %) может быть выбрана каждая. Одно решение заключается в том, чтобы взять сумму обратных значений коэффициентов, и исходя из этого вычислять проценты.

Таблица 3: Вероятность оказаться родителем

Хромосома	Подходящесть
1	(1/84)/0.135266 = 8.80%
2	(1/24)/0.135266 = 30.8%
3	(1/26)/0.135266 = 28.4%
4	(1/133)/0.135266 = 5.56%
5	(1/28)/0.135266 = 26.4%

Для выбора 5-и пар родителей (каждая из которых будет иметь 1 потомка, всего - 5 новых решений), представим, что у нас есть 10000-гранная игральная кость, на 880 сторонах отмечена хромосома 1, на 3080 - хромосома 2, на 2840 сторонах - хромосома 3, на 556 - хромосома 4 и на 2640 сторонах отмечена хромосома 5. Чтобы выбрать первую пару кидаем кость два раза и выбираем выпавшие хромосомы. Таким же образом выбирая остальных, получаем:

Таблица 4: Симуляция выбора родителей

Хромосома отца	Хромосома матери
3	1
5	2
3	5
2	5
5	3

Каждый потомок содержит информацию о генах и отца и от матери. Вообще говоря, это можно обеспечить различными способами, однако в нашем случае можно использовать т.н. "кроссовер" (cross-over). Пусть мать содержит следующий набор решений: a1,b1,c1,d1, а отец - a2,b2,c2,d2, тогда возможно 6 различных кросс-оверов (| = разделительная линия):

Таблица 5: Кросс-оверы между родителями

Хромосома-отец	Хромосома-мать	Хромосома-потомок
$a_1 \mid b_1,c_1,d_1$	$a_2 \mid b_2,c_2,d_2$	a_1,b_2,c_2,d_2 or a_2,b_1,c_1,d_1
$a_1,b_1 \mid c_1,d_1$	$a_2,b_2 \mid c_2,d_2$	a_1,b_1,c_2,d_2 or a_2,b_2,c_1,d_1
$a_1,b_1,c_1 \mid d_1$	$a_2,b_2,c_2 \mid d_2$	a_1,b_1,c_1,d_2 or a_2,b_2,c_2,d_1

Таблица 6: Симуляция кросс-оверов хромосом родителей

Хромосома-отец	Хромосома-мать	Хромосома-потомок
(13 | 5,7,3)	(1 | 28,15,3)	(13,28,15,3)
(9,13 | 5,2)	(14,9 | 2,4)	(9,13,2,4)
(13,5,7 | 3)	(9,13,5 | 2)	(13,5,7,2)
(14 | 9,2,4)	(9 | 13,5,2)	(14,13,5,2)
(13,5 | 7, 3)	(9,13 | 5, 2)	(13,5,5,2)

Таблица 7: Коэффициенты выживаемости потомков (fitness)

Хромосома-потомок	Коэффициент выживаемости
(13,28,15,3)	|126-30|=96
(9,13,2,4)	|57-30|=27
(13,5,7,2)	|57-30|=22
(14,13,5,2)	|63-30|=33
(13,5,5,2)	|46-30|=16

Средняя приспособленность (fitness) потомков оказалась 38.8, в то время как у родителей этот коэффициент равнялся 59.4. Следующее поколение может мутировать. Например, мы можем заменить одно из значений какой-нибудь хромосомы на случайное целое от 1 до 30.

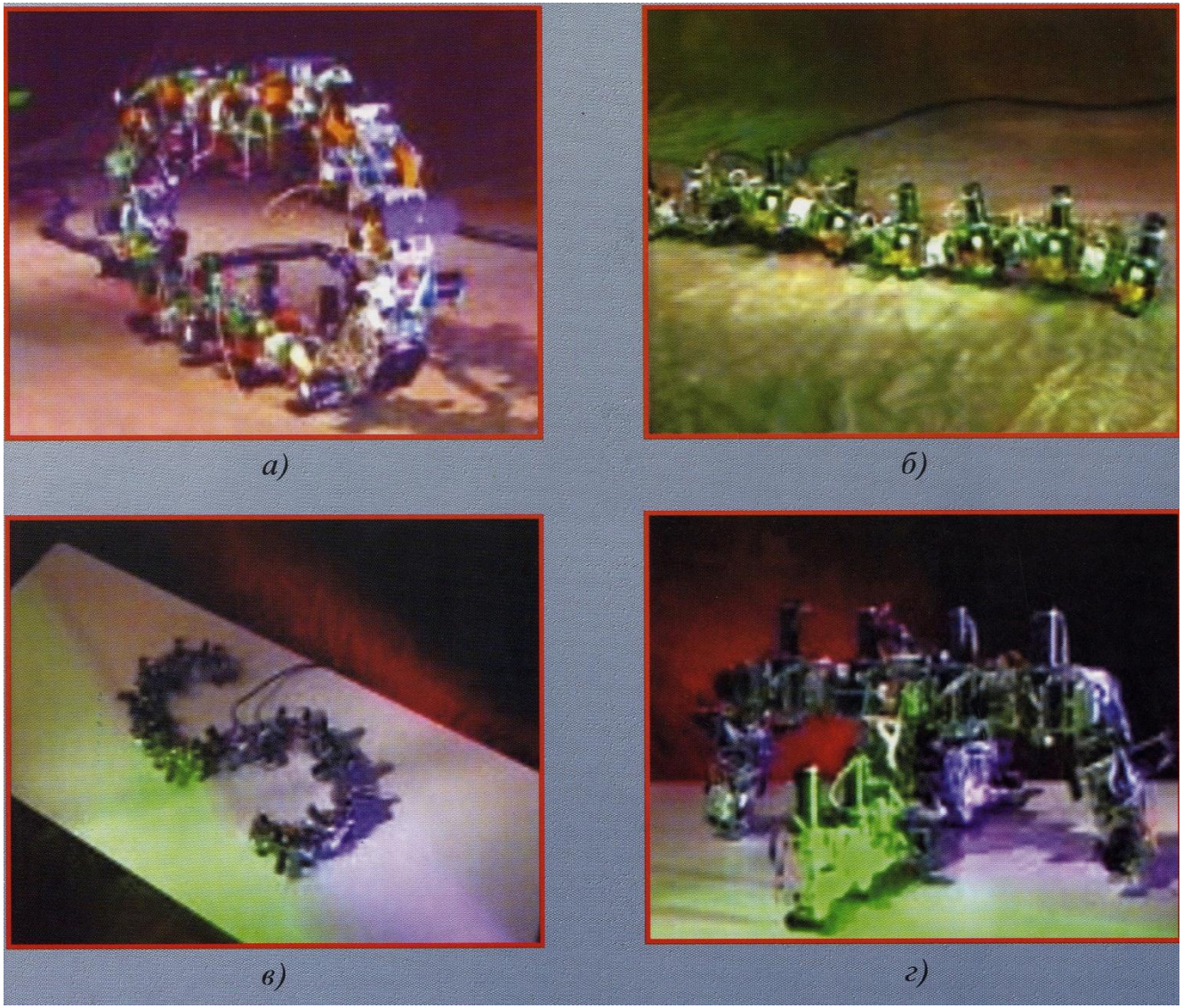
Продолжая таким образом, одна хромосома в конце концов достигнет коэффициента выживаемости 0, то есть станет решением.

Применение ГА

Генетические алгоритмы применяются при разработке программного обеспечения, в системах искусственного интеллекта, оптимизации, искусственных нейронных сетях и в других отраслях знаний. Следует отметить, что с их помощью решаются задачи, для которых ранее использовались только нейронные сети. В этом случае генетические алгоритмы выступают просто в роли независимого от нейронных сетей альтернативного метода, предназначенного для решения той же самой задачи.
Генетические алгоритмы часто используются совместно с нейронными сетями. Они могут поддерживать нейронные сети или наоборот, либо оба метода взаимодействуют в рамках гибридной системы, предназначенной для решения конкретной задачи. Генетические алгоритмы также применяются совместно с нечеткими системами.

Использование генетических алгоритмов для автоматического формирования программ управления движением автономных реконфигурируемых мехатронно-модульных роботов

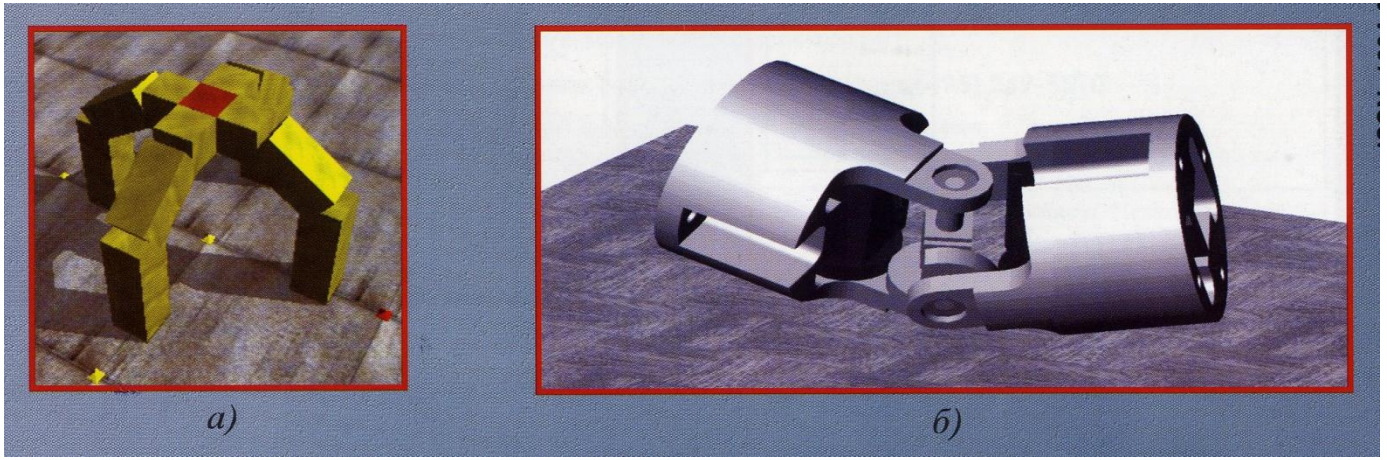
Совершено самостоятельный аспект исследований, которые активно проводятся в рамках этой очень широкой тематики, связан с попытками применения генетических алгоритмов для решения задач по автоматическому формированию программ, выполняющих требуемые функции. Подобные задачи приобретают особую актуальность в контексте проблем разработки принципиально нового класса технических систем, обладающих способностями к самоорганизации и самообучению. Одним из примеров систем такого рода являются многозвенные реконфигурируемые мехатронно-модульные роботы, техническая реализуемость которых подтверждается результатами ряда известных проектов. Так, в лабораторных испытаниях конкретных образцов реконфигурируемых мехатронно-модульных роботов различных типов, включая *Poly Bot (PARK, USA), Polypod (Stanford University, USA), MTRAN (AIST, Japan)* и др., неоднократно демонстрировалась возможность изменения кинематической структуры за счет автоматической перестыковки ее фрагментов.



PolyBot (PARK, Xerox, USA)

Многозвенные реконфигурируемые мехатронно-модульные роботы в зависимости от условий своего функционирования и специфики решаемых задач должны не только автоматически изменять свою структуру, но и синтезировать необходимые программы управления.

Рассмотрим на примере обучения шагающего устройства с четырьмя конечностями, в основу кинематической схемы которого положены типовые мехатронные модули с двухстепенными шарнирами вращения



Формирование программы управления мехатронно-модульного робота в конфигурации шагающего устройства предполагает необходимость построения целесообразной последовательности циклических изменений состояния конечностей, обеспечивающей тот или иной вид походки. При этом изменение состояния конечностей соответствует выполнению элементарных движений из следующего набора:

- поднять конечность;
- опустить конечность;
- повернуть конечность вперед (+15°);
- повернуть конечность в нейтральное положение (0°);
- повернуть конечность назад (—15°).

Будем считать, что реализация оговоренных движений определяется радом допущений:

- прямоугольная система координат робота связана с его узловым модулем;
- точки крепления конечностей к узловому модулю совпадают с осями системы координат робота;
- движение каждой конечности в вертикальной, плоскости осуществляется поворотами шарниров первого и второго модулей (в порядке следования от узлового)
- движение каждой конечности в горизонтальной плоскости осуществляется поворотом шарнира, первого модуля (по отношению к узловому)

- Структура хромосомы, отвечающей этим требованиям, разбивается на несколько фрагментов, каждый из которых будет кодировать один такт алгоритма управления, задающий изменение состояний конечностей робота. Для кодирования могут использоваться три бита, первый из которых устанавливает движение конечности в вертикальной плоскости, а второй и третий — в горизонтальной плоскости

№ бита	Значение	Элементарное движение
1	0 1	Опустить конечность Поднять конечность
2, 3	00 01 10 11	Повернуть конечность в нейтральное положение Повернуть конечность вперед Повернуть конечность назад Оставить конечность в текущем положении

Оценка полезности хромосом в данном случае сводится к сравнению эффективности отдельных циклов представляемых ими алгоритмов управления походкой шагающего робота. Очевидным критерием эффективности исследуемых алгоритмов управления, шагающим роботом является значение его перемещения в результате отработки одного цикла.

Следовательно, исходя из предположения, что движение робота при моделировании начинается в точке с нулевыми координатами, функция оценки полезности хромосом может быть записана в следующем виде:

$$f = x^2+y^2$$

где *x, y* — координаты положения робота после отработки одного цикла синтезированного алгоритма управления движением.

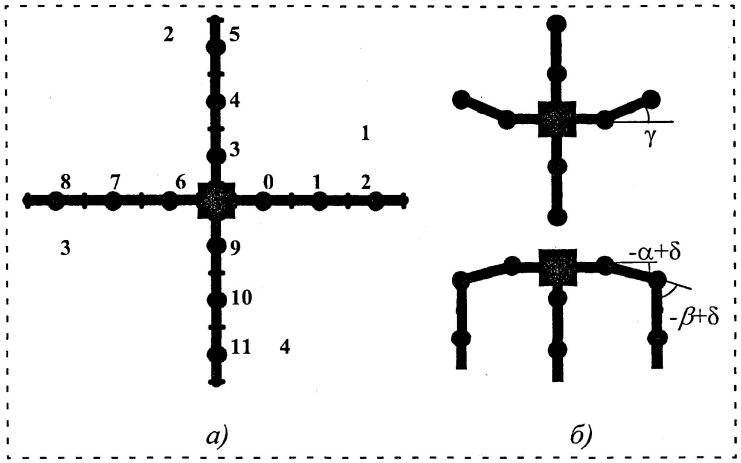


Рис. 10. Нумерация модулей (*а*) и движения конечностей (*б*) многозвенного мехатронно-модульного шагающего робота

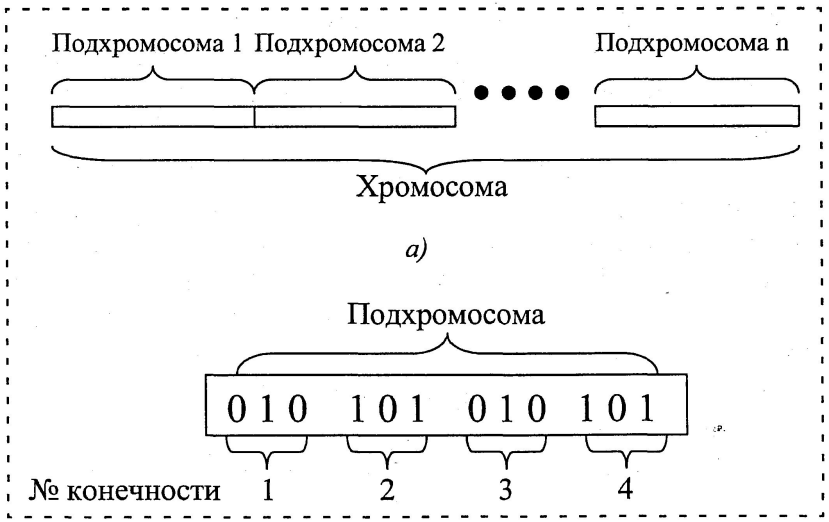


Рис. 11. Структура хромосомы и ее фрагментов, используемых при автоматическом синтезе алгоритмов управления мехатронно-модульным шагающим роботом для кодирования движений его конечностей

Машинная реализация процесса эволюции особей, которые представляются в виде хромосом с выбранным способом структуризации, обусловливает необходимость модификации общепринятых схем выполнения генетических операций. Так, анализ специфики решаемой задачи с учетом особенностей построения хромосом показал целесообразность замены традиционной схемы выполнения операции скрещивания, при которой осуществляется взаимный обмен одноименными генами родительской пары хромосом фиксированной длины, двумя новыми вариантами.

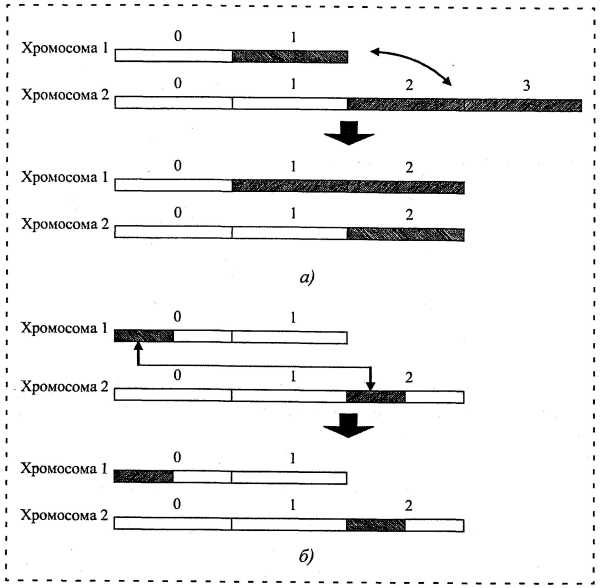


Рис. 12. Выполнение операций скрещивания хромосом в задачах автоматизированного синтеза алгоритмов управления движением конечностей мехатронно-модульного шагающего робота

Первый из них сводится к обмену хромосом своими составными частями, начиная от произвольно вы бираемых точек разрыва до конца битовых строк
Если при этом длина какого-либо из потомков превышает максимально возможную, то он усекается до требуемого размера.
Второй вариант операции скрещивания заключается в обмене родительских особей одинаковым набором одноименных генов из состава подхромосом. По характеру выполнения эта операция может обеспечивать как одноточечное, так и многоточечное скрещивание.

Как известно, формирование новой популяции особей в процессе их эволюции осуществляется в результате рекомбинации отобранных хромосом с помощью не только операций скрещивания, но и мутации.

Применительно к задачам автоматизации синтеза алгоритмов управления движением мехатронно-модульного шагающего робота методами эволюционного программирования выполнение операции мутации хромосом также должно осуществляться в нескольких различных вариантах.

Первый вариант связан с реализацией классической схемы, предполагающей изменение некоторого числа битов хромосомы на противоположные по значению. При этом число битов зависит от вероятности выполнения данного оператора, которая задается в качестве параметра. Второй вариант операции мутации заключается в случайном изменении местоположения подхромосом при сохранении исходного размера битовой строки.

И, наконец, третий вариант операции мутации обеспечивает дополнительное подключение случайно сгенерированной подхромосомы в состав структуры имеющейся битовой строки с увеличением ее длины

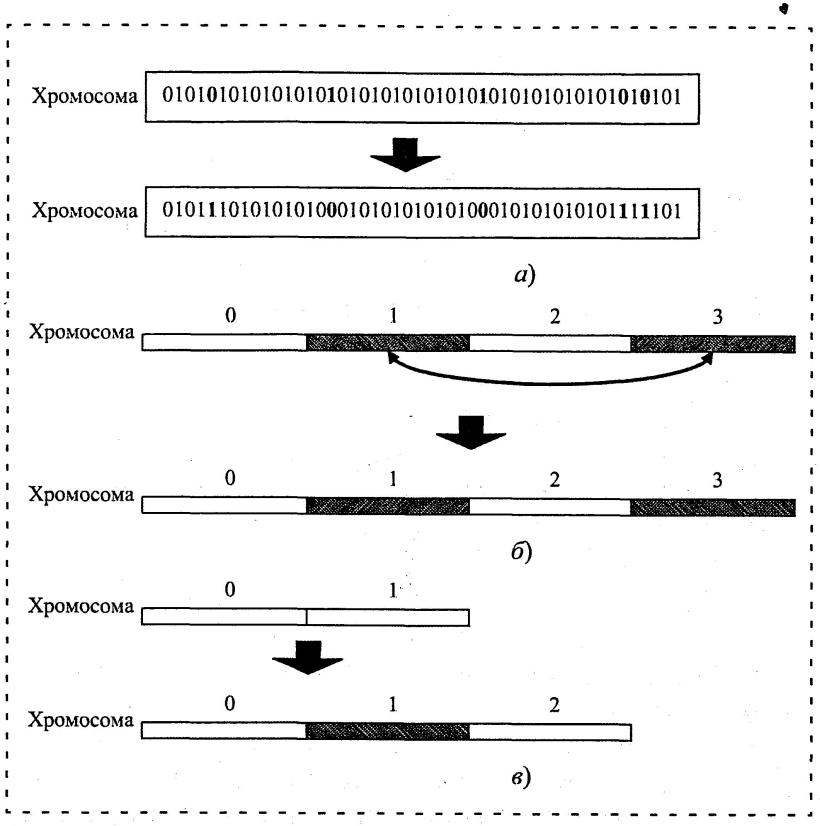
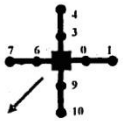
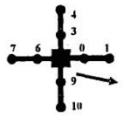
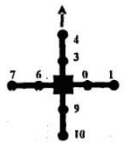
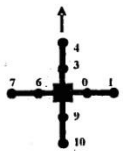
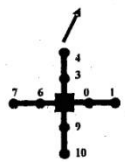
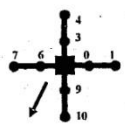
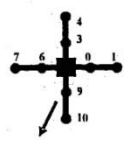


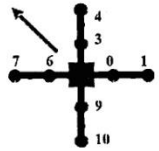
Рис. 13. Выполнение операций мутации хромосом в задачах автоматизированного синтеза алгоритмов управления движением конечностей мехатронно-модульного шагающего робота

В ходе выполнения экспериментальных исследований селекция синтезируемых хромосом осуществлялась по методу элитного отбора. Для эмуляции движений робота, отвечающих кодам полученных хромосом, использовался специализированный комплекс программно- инструментальных средств моделирования виртуальной реальности **ODE (Open Dynamic Engine).** При этом продолжительность "жизни" каждой особи ограничивалась 40 тактами, где под тактом понимается период выполнения команд одной подхромосомы.
Для полноты исследований был проведен сравнительный анализ программ управления роботом, одна из которых была синтезирована высококвалифицированным экспертом, а остальные формировались в автоматическом режиме с помощью генетического алгоритма. Составленная экспертом программа на алгоритмическом уровне рассмотрения представляла собой последовательность элементарных движений конечностей, регулярное повторение которой должно обеспечивать реализацию соответствующей походки робота.

Результаты экспериментов по оценке эффективности программ управления походкой шагающего робота, синтезированных с помощью генетического алгоритма

№ опыта	Хромосома	Число подхромосом	Пригод-ность	Вектор направления
1	000110110100 111100010111 110011111101	3	1,7961	
2	010101111010 001100000000 011100010101 110000111001	4	2,3882	
3	001000110010 011100101101 001101111000 110010010001	4	2,4966	
4	011100101100 110010010001	2	2,614	
5	011110101100 110010010001	2	2,7276	
6	001001010011 110010000010 101110100111	3	2,5102	
7	110001010011 111010000010 101110100111	3	2,8601	

Результаты эксперимента по оценке эффективности программы управления походкой шагающего робота, синтезированной экспертом

№ опыта	Хромосома	Число подхромосом	Пригод-ность	Вектор направления
1	101011110011 111010111001 011011011011 011101011110 010111001111 011011011011	6	2,5396	

Анализ экспериментальных данных показывает, что в смысле полезности полученных хромосом автоматически формируемые с помощью генетического алгоритма решения в ряде отдельных случаев оказываются намного более эффективными, чем синтезируемые экспертом. При этом следует отметить, что если поиск приемлемых решений по формированию программ управления походкой шагающего робота в автоматическом режиме по времени занимает от 10 до 30 минут, то для эксперта решение аналогичной задачи требует от 1,5 до 3 суток.

АППАРАТНАЯ РЕАЛИЗАЦИЯ ГЕНЕТИЧЕСКИХ АЛГОРИТМОВ

Идея применения генетических алгоритмов в системах автоматизированного проектирования активно развивается наряду с другими направлениями. Впервые эта идея была предложена С. Луисом [Louis et al., 1991] и Д. Раулинсом [Rawlins et al., 1993] в 1991 году и экспериментально проверена в области цифровых схем. В дальнейшем, предложенные методы были развиты и доработаны, и получили применение во многих автоматизированных системах проектирования аппаратуры. Использование генетических алгоритмов (ГА) [Курейчик 2004 и др.] как механизма для автоматического проектирования схем на реконфигурируемых платформах [Blondet et al., 2003], получило название эволюционные аппаратные средства (Evolvable Hardware) [Higuchi et al., 1993], [Sakanashi et al., 1999], которое также используется синонимом для несколько общего направления, известного как эволюционная электроника (Evolutionary Electronics) [Zebulum et al., 2002].

Для автономных решений и задач, связанных с построением эволюционных аппаратных средств, программная реализация ГА является неприемлемой по целому ряду критериев. Сам факт автономности исключает наличие возможности использования программных решений, выполняемых на ПК или кластерным методом. С другой стороны, автономные системы, как правило, функционируют в режиме реального времени, что накладывает ряд требований на временные характеристики используемых алгоритмов, в связи с чем, вопрос использования программных моделей перестает быть актуальным.

Общее повышение быстродействия

Compact Genetic Algorithm	Параметры	Время
Программная реализация	200 МГц, Ultra Sparc 2	2:30 мин.
Аппаратная реализация	20 МГц, FPGA	0.15 сек
Увеличение быстродействия	**1 000 раз**	
Univariate Marginal Distributiona Algorithm (UMDA)	Параметры	Время
Программная реализация	540 МГц, P3	23 сек
Аппаратная реализация	125 МГц, FPGA	84 мк.сек.
Увеличение быстродействия	**27 380 раз**	