УДК 004.023

ГЕНЕТИЧЕСКИЕ АЛГОРИТМЫ НА ПРИМЕРАХ РЕШЕНИЯ ЗАДАЧ РАСКРОЯ

А.В. Подлазова

Московский государственный институт стали и сплавов (технологический университет), г. Москва

Рассматривается решение задач дискретной оптимизации с помощью генетических алгоритмов. Описываются созданные для их применения структуры данных, приводятся разработанные генетические операторы — важные составляющие генетических алгоритмов. Обсуждаются общие характеристики генетических алгоритмов, достоинства и недостатки.

ВВЕДЕНИЕ

Идея использования *принципов биологической эволюции* для решения оптимизационных задач возникала в различных модификациях у ряда авторов. Первые публикации на эту тему появились в 1960-х г. А в 1975 г. вышла основополагающая книга Холланда «Адаптация в естественных и искусственных системах», в которой и был предложен собственно первый *генетический алгоритм* [1].

Сегодня о генетических алгоритмах можно говорить как о методе, которым решено множество различных задач [2—24]. Об этом свидетельствует огромная библиография, в том числе и на русском языке. Тем не менее, генетические алгоритмы не перестают быть предметом споров об эффективности их работы и целесообразности их использования.

В данной работе приводится формальная постановка задачи, на решение которой изначально был направлен первый генетический алгоритм, и общая схема работы самого алгоритма. Описываются составляющие генетического алгоритма и их назначение. Для решения конкретной задачи разработчик, исследователь может сконструировать свою конкретную схему работы генетического алгоритма из его составляющих. При этом разработчик руководствуется особенностями предметной области, формализации задачи, структурой используемых данных, возможно, даже результатами тестирования других схем генетического алгоритма. Разумеется, в строении всех таких алгоритмов есть общие элементы и последовательность действий, которые являются основой генетических алгоритмов и отличают их от многих других. Данная статья посвящена опыту конструирования конкретных вариантов генетического алгоритма.

1. ФОРМАЛИЗАЦИЯ ГЕНЕТИЧЕСКОГО АЛГОРИТМА

Генетический алгоритм — это математическая модель эволюции популяции искусственных особей. Формализация постановки задачи, на решение которой направлен алгоритм, такова.

В популяции каждая особь k характеризуется своей *хромосомой* S_k. Формально, хромосома есть цепочка символов $S_k = (S_{k1}, S_{k2}, ..., S_{kN})$, где N — длина цепочки. Хромосома определяет *приспособленность* (*пригодность*) особи $f_k = f(S_k)$; $k = 1, 2, ..., n$, n — численность *популяции*. Цель состоит в максимизации *функции приспособленности* $f(S_k)$, т. е. в нахождении особи с максимальной приспособленностью. Важно отметить, что генетический алгоритм ищет решение, как можно более близкое к оптимуму, но не гарантирует нахождение точного максимума функции приспособленности.

Эволюция популяции моделируется последовательностью *поколений* $\{S_k(t)\}$, $t = 0, 1, 2, ...$ В каждый следующий момент времени его состав меняется с целью увеличения пригодности особей. Это и есть процесс *эволюции* популяции.

Для каждого следующего поколения отбираются особи с относительно б**о**льшими значениями приспособленностей. Хромосомы приспособленных особей «скрещиваются» и подвергаются малым «мутациям». Отбор, скрещивание, мутация — все это является *генетическими операторами*, процедурами, применение которых позволяет получить новое поколение популяции.

Формально, *генетический алгоритм* — это любой алгоритм поиска приближенного решения поставленной задачи, построенный по следующей схеме, *схеме генетического алгоритма* (t-е поколение популяции обозначается как $\{S_k(t)\}$) [2, 3]:

Шаг 0. Создать случайную начальную популяцию $\{S_k(0)\}$.

Шаг 1. Вычислить приспособленность $f(S_k)$ каждой особи S_k популяции $\{S_k(t)\}$.

Шаг 2. Производя отбор особей из $\{S_k(t)\}$ в соответствии с их приспособленностями $f(S_k)$ и применяя генетические операторы (скрещивания и мутации) к отобранным особям для получения потомства, сформировать популяцию следующего поколения $\{S_k(t+1)\}$.

Шаг 3. Повторять шаги 1, 2 для $t = 0, 1, 2, \ldots$ до тех пор, пока не выполнится некоторое условие окончания эволюционного поиска (прекращается рост максимальной приспособленности в популяции, число поколений t достигает заданного предела и т. п.).

Возможны различные варианты генетического алгоритма, которые отличаются по схемам отбора особей из текущего поколения в новое, операторам скрещивания и мутации хромосом особей, по форме представления хромосом и т. д. Традиционный вид генетического алгоритма базируется на следующей частной схеме [2, 3]:

1) цепочки символов в хромосомах бинарные (символы S_{ki} принимают значения 0 либо 1), длина цепочек постоянна ($N = \text{const}$);

2) метод отбора — пропорционально-вероятностный (см. далее);

3) скрещивания производятся по схеме одноточечного скрещивания (см. далее).

Схемы отбора. Пропорционально-вероятностный отбор означает, что отбор особей производится с вероятностями q_k, которые пропорциональны приспособленностям f_k особей $q_k = f_k / \sum_l f_l$. Эту схему отбора можно представить как выбор особи с помощью рулетки, относительные площади секторов которой равны *вероятностям* $q_k = f_k / \sum_l f_l$.

Например, если текущее поколение состоит их 4-х особей S_1, S_2, S_3 и S_4 со значениями пригодности $f_1 = 2, f_2 = 4, f_3 = 1, f_4 = 1$, то в формировании нового поколения они будут участвовать с вероятностями соответственно 0,25, 0,5, 0,125 и 0,125 (рис. 1).

Чем выше пригодность особи (чем она «сильнее»), тем больше у нее шансов оставить потомство. По аналогии с реальной жизнью четыре раза вращается рулетка и четыре раза выбирается особь для скрещивания или мутации. Одна и та же особь может участвовать в формировании потомства несколько раз, вероятнее всего — это будет более приспособленная особь.

Известны и другие методы отбора. Например, отбор может быть *ранжированным*: все особи ранжируются, т. е. упорядочиваются по приспособленностям, и заданная часть лучших особей (например, лучшая половина) отбирается для формирования следующего поколения.

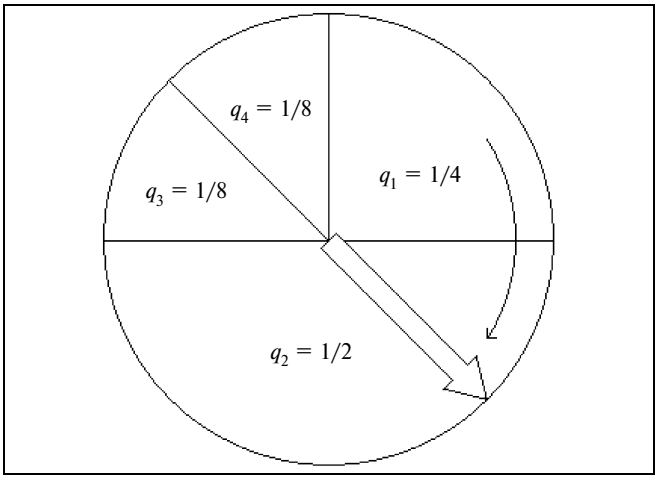

Рис. 1. Пример схемы отбора для $n = 4$ и $f(S_1) = 2$, $f(S_2) = 4$, $f(S_3) = 1$, $f(S_4) = 1$

Схемы скрещивания. Одноточечное скрещивание организуется следующим образом. Если есть два родителя с хромосомами $S_1 = (S_{11}, S_{12}, \ldots, S_{1N})$ и $S_2 = (S_{21}, S_{22}, \ldots, S_{2N})$, то хромосомы их потомков в новом поколении есть $(S_{11}, \ldots, S_{1m}, S_{2m+1}, \ldots, S_{2N})$ и $(S_{21}, \ldots, S_{2m}, S_{1m+1}, \ldots, S_{1N})$; т. е. «голова» и «хвост» хромосомы потомка берутся от разных родителей. Точка скрещивания выбирается случайным образом, в приведенном примере она располагается между m-м и $(m+1)$-м символами. Аналогичным образом может быть организовано *двухточечное* и *«несколько-точечное» скрещивание.*

Существует также понятие *инверсии*, это изменение порядка следования символов в участках хромосом. В некоторых схемах генетических алгоритмов используется *равномерное скрещивание.* Это означает, что два родителя имеют двух потомков, причем символы хромосомы одного из потомков выбираются случайно от любого из двух родителей (но с сохранением порядка следования символов), а второму потомку достаются оставшиеся символы. Например, два потомка родителей $S_1 = (S_{11}, S_{12}, \ldots, S_{1N})$ и $S_2 = (S_{21}, S_{22}, \ldots, S_{2N})$ могут иметь следующие хромосомы $(S_{11}, S_{22}, S_{13}, S_{14}, S_{25}, \ldots, S_{2N})$ и $(S_{21}, S_{12}, S_{23}, S_{24}, S_{15}, \ldots, S_{1N})$.

Известные схемы описанных типов генетических операторов — отбора, скрещивания, мутации — имеют множество разновидностей [2, 3]. Приведенные выше варианты дают представление о внутреннем устройстве генетических алгоритмов вообще и облегчают понимание конкретных реализаций метода, представленых в статье.

2. ДОСТОИНСТВА И НЕДОСТАТКИ ГЕНЕТИЧЕСКИХ АЛГОРИТМОВ

Перечислим признанные достоинства и недостатки известных реализаций генетического алгоритма.

Достоинства:

- широкая область применения генетических алгоритмов [4, 5];
- возможность проблемно-ориентированного кодирования решений, подбора начальной популяции, комбинирования генетических алгоритмов с даже неэволюционными алгоритмами, продолжения процесса эволюции до тех пор, пока имеются необходимые ресурсы [6];
- пригодность для поиска в сложном пространстве решений большой размерности [7, 8];
- отсутствие ограничений на вид целевой функции (функции приспособленности) [9];
- ясность схемы построения генетических алгоритмов;
- интегрируемость с искусственными нейросетями и нечеткой логикой [9].

Недостатки:

- эвристический характер генетических алгоритмов не гарантирует оптимальности полученного решения (правда, на практике зачастую важно за заданное время получить одно или несколько рациональных решений, более или менее приближающихся к оптимальным; к тому же исходные данные в задаче могут динамически меняться, быть неточными или неполными);
- относительно высокая вычислительная трудоемкость, причинами которой является то обстоятельство, что в ходе моделирования эволюции многие решения отбрасываются как неперспективные (тем не менее, многочисленные экспериментальные данные указывают на то, что временная сложность генетических алгоритмов в среднем ниже, чем у лучших конкурирующих алгоритмов, но не более, чем на один порядок) [10];
- невысокая эффективность генетических алгоритмов на заключительных фазах моделирования эволюции; это объясняется тем, что генетические операторы не ориентированы на быстрое попадание в локальный оптимум.

Кроме того, анализ результатов использования генетических алгоритмов [11] позволяет выделить следующие признаки, при наличии которых задача решается генетическим алгоритмом более эффективно:

- большое пространство поиска, ландшафт которого является негладким (содержит несколько экстремумов);
- сложно формализуемая функция оценки качества решения (функция пригодности);
- многокритериальность поиска;
- поиск по заданным критериям приемлемого, а не единственного оптимального решения;
- многопараметричность задачи;
- поиск решения в режиме реального времени или близком к нему.

При использовании традиционных алгоритмов многопараметрического поиска для решения различных задач возникает ряд трудностей, к которым относятся [12]:

- резкий рост вычислительных затрат и времени поиска при увеличении числа варьируемых параметров («проклятие размерности»);
- локальный характер алгоритмов поиска, связанный с необходимостью вычисления производных (градиента) целевой функции на каждом шаге поиска;
- возможность «зависания» алгоритма поиска в окрестности одного из локальных экстремумов;
- низкая помехозащищенность алгоритма;
- низкая эффективность поиска при наличии «овражных» ситуаций.

Привлекательность генетических алгоритмов состоит именно в том, что они позволяют в значительной мере преодолеть указанные трудности.

3. ПОСТРОЕНИЕ РАЦИОНАЛЬНОГО ПЛАНА РАСКРОЯ

Рассматриваемая задача — поиск рационального плана раскроя плоского объекта на предметы различной формы. Задачи такого рода впервые были поставлены еще в 1940-х гг. академиком Л.В. Канторовичем [25]. С тех пор появилось большое количество новых постановок и методов решения [13]. Однако существуют практически значимые постановки задач и технологические ограничения, для которых решение задачи раскроя и разработка новых методов решения по-прежнему актуальны [13—15].

Вообще, задача плоского раскроя — это оптимизационная задача поиска наиболее плотного размещения множества меньших по размеру плоских *предметов*, деталей, на больших *объектах*, заготовках. В Московском институте стали и сплавов (МИСиС) задача плоского раскроя была неформально поставлена следующим образом: имеется сырье, прямоугольные листы металла, которые требуется раскраивать с максимально возможной экономией материала. Но при этом одновременно были поставлены две разные задачи раскроя:

- для лазерного технологического участка научно-исследовательской лаборатории процессов пластической деформации и упрочнения поставлена задача раскроя прямоугольных листов на прямоугольные предметы разного размера в заданном количестве (так называемая задача прямоугольного раскроя);

• кафедра обработки металлов давлением поставила задачу раскроя набора прямоугольных листов на круглые заготовки разного размера в заданном количестве (так называемая задача круглого раскроя).

Требуется подобрать алгоритм, с помощью которого можно эффективно решить обе задачи, не углубляясь в два принципиально различных исследования, и «не очень много теряя» из-за универсальности подхода к решению. Был выбран генетический алгоритм по следующим причинам:

во-первых, задача раскроя любого вида является NP-полной [25];

во-вторых, время получения решения для заказчиков предполагается порядка десятка секунд для размерности 100, это достаточно высокие требования ко времени;

в-третьих, один из их выдающихся результатов коллектива профессора Уфимского государственного авиационного технического университета Э.А. Мухачевой [13] — это внушительное по объемам исследование вопросов прямоугольного раскроя, которое показало, что в среднем генетические алгоритмы в различных модификациях дают лучшие результаты в сравнении другими методами.

Уже существуют подходы к решению задач рационального раскроя на предметы произвольной формы [26]. К сожалению, их неэффективно использовать в случаях, вырожденных до простых геометрических форм. Значительные средства, введенные в алгоритм [26] для обработки произвольности формы, прокручиваются вхолостую, занимая при этом все те же вычислительные мощности.

Было принято решение попробовать сконструировать некоторый универсальный подход к решению задач плоского раскроя на основе генетического алгоритма.

Итак, известная математическая постановка задачи прямоугольного раскроя (рис. 2) как задачи оптимизации такова. Заданы полубесконечная полоса ширины W (большой объект, на котором размещаются предметы), m прямоугольных предметов, длина и ширина которых известна как (l_i, w_i), $i = 1, 2, ..., m$, и условия, где (x_i, y_i) — координаты левого нижнего угла прямоугольника i на полосе:

— при размещении на полосе никакие два предмета не пересекаются друг с другом $((x_i \geqslant x_j + l_j) \vee (x_j \geqslant x_i + l_i)) \vee ((y_i \geqslant y_j + w_j) \vee (y_j \geqslant y_i + w_i))$ для $i, j = 1, 2,..., m$, $i \neq j$;

— никакой предмет не пересекает границ полосы $(x_i \geqslant 0) \wedge (y_i \geqslant 0) \wedge (y_i + w_i \leqslant W)$ для $i = 1, 2, ..., m$.

Требуется так разместить на полубесконечной полосе набор прямоугольных предметов без перекрытий, чтобы занятая ими часть полосы была минимальна по длине; т. е., формально требуется найти такой набор (x_i, y_i), $i = 1, 2, ..., m$, чтобы $L = \max\limits_i (x_i + l_i) \to \min$. Геометрический смысл переменных W, L, (l_i, w_i), (x_i, y_i), $i = 1, 2, ..., m$ проиллюстрирован на рис. 3.

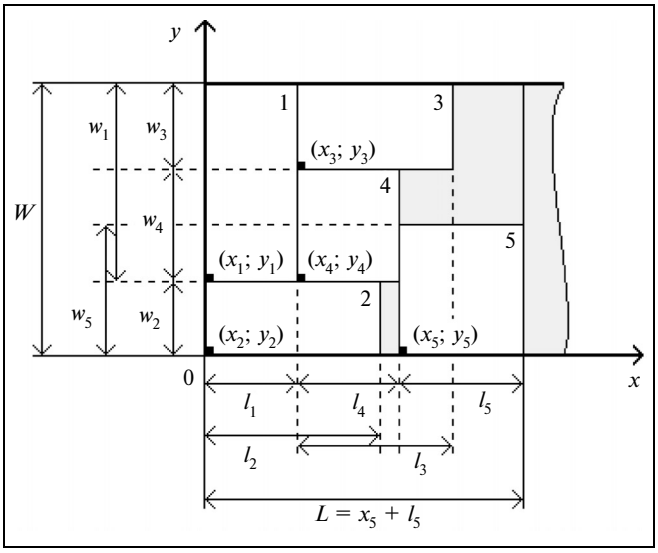

Рис. 2. Задача прямоугольного раскроя

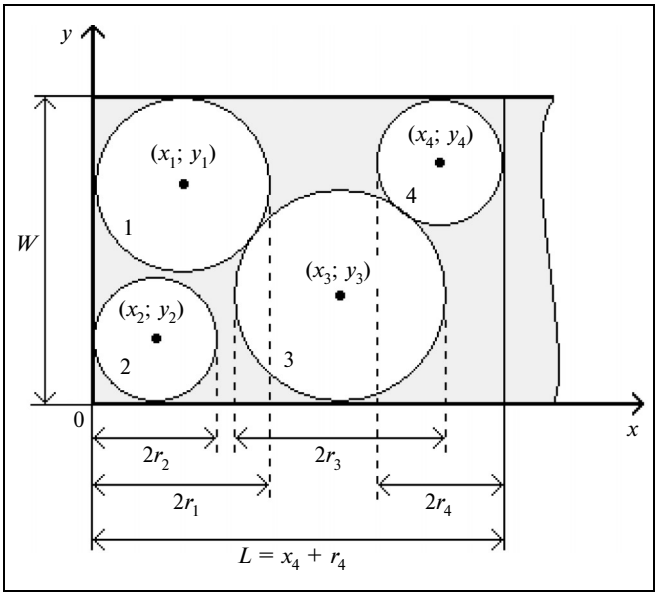

Рис. 3. Задача круглого раскроя

Предлагаемая математическая постановка задачи круглого раскроя (рис. 3) такова. Заданы полубесконечная полоса ширины W, m круглых предметов, радиусы которых известны как r_i, $i = 1, 2, ..., m$ и условия, где (x_i, y_i) — координаты центра окружности i на полосе:

— при размещении на полосе никакие два предмета не пересекаются друг с другом $(x_i - x_j)^2 + (y_i - y_j)^2 \geqslant (r_i + r_j)^2$ для $i, j = 1, 2, ..., m$, $i \neq j$;

— никакой предмет не пересекает границ полосы $(x_i - r_i \geqslant 0) \wedge (y_i - r_i \geqslant 0) \wedge (y_i + r_i \leqslant W)$ для $i = 1, 2, ..., m$.

Требуется так разместить на полубесконечной полосе набор круглых предметов без перекрытий, чтобы занятая ими часть полосы была минимальна по длине; т. е., формально требуется найти такой набор (x_i, y_i), $i = 1, 2, ..., m$, чтобы $L = \max\limits_i (x_i + r_i) \rightarrow \rightarrow \min$. Геометрический смысл переменных W, L, r_i, (x_i, y_i), $i = 1, 2, ..., m$ проиллюстрирован на рис. 3.

Для реализации генетического алгоритма особью предлагается считать любое размещение заданных предметов на полосе, удовлетворяющее условию задачи. Пусть все предметы некоторым образом пронумерованы. Тогда пусть хромосома — это перечисление всех номеров предметов в некотором порядке, отражающем их размещение. Физический внешний вид особи, конкретные координаты размещения предметов на полосе, можно получить, выложив предметы на полосу в порядке их перечисления в хромосоме по некоторым правилам переработки — декодирования. Эти правила называются процедурой декодирования, или декодером. Различные декодеры могут работать по разной схеме, но суть одна — выложить предметы на полосу согласно некоторому правилу в порядке, указанном в хромосоме. Правила являются эвристическими и содержательно строятся так, чтобы размещение было по возможности плотным.

В качестве критерия оптимизации, критерия оценки пригодности особи в поставленных задачах рассматривается длина полосы, занятая размещением предметов.

На рис. 4 и рис. 5 проиллюстрированы примеры результата работы двух разных декодеров для особи с хромосомой (4, 2, 5, 1, 3).

Для задачи прямоугольного раскроя коллективом Э.А. Мухачевой разработано понятие блок-структуры, с помощью которой удобно хранить информацию о размещении и рационально располагать каждый следующий предмет, учитывая ранее образовавшиеся пустоты. На рис. 6 приведен пример размещения, имеющего структуру из пяти блоков, каждый из которых состоит из участков нескольких предметов и пустот.

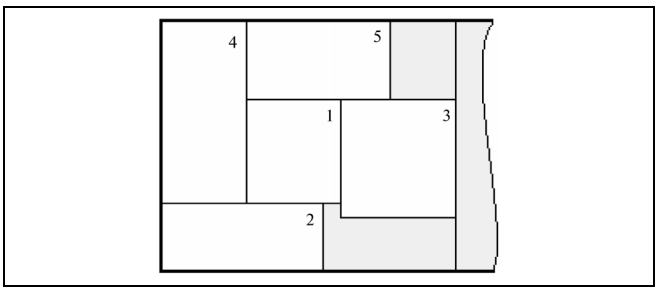

Рис. 4. Размещение прямоугольных предметов

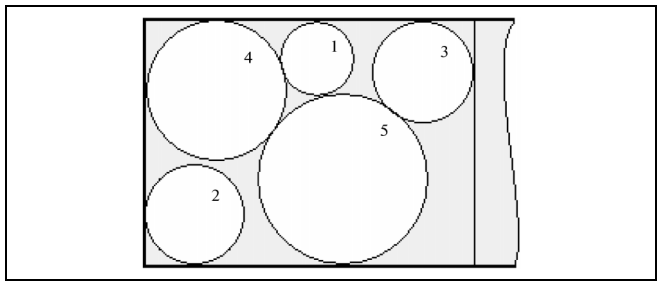

Рис. 5. Размещение круглых предметов

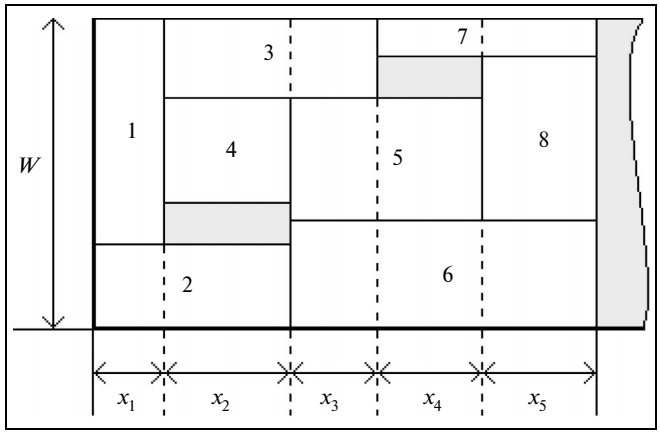

Рис. 6. Схема блок-структуры

Пустоты учитываются и нумеруются, начиная с числа, на единицу большего количества предметов, но со знаком минус. Декодер, работающий с такой структурой данных, называется блочным [13]. На рис. 6 размещение представлено в виде совокупности кортежей (блоков): $(1_j, 2_j, ...)x_j$, где $j = 1, 2, ...$ А именно: $(1; 2)x_1$, $(3; 4; -9; 2)x_2$, $(3; 5; 6)x_3$, $(7; -10; 5; 6)x_4$, $(7; 8; 6)x_5$, $(-11)x_6$. Длина занятой части полосы $L = \sum\limits_{j=1}^{5} x_j$. Одна из особенностей блочного декодера в том, что размещая очередной предмет, он сначала проверяет все образовавшиеся ранее пустоты, подойдут ли они по размеру, для достижения большей плотности размещения.

Для задачи круглого раскроя был спроектирован специальный краевой декодер [15]. Краем считается неразрывная цепочка касающихся друг друга либо сторон полосы предметов, разделяющая все остальные предметы и незанятую часть полосы (рис. 7). Край размещения инициализируется последовательностью вида: {*Lt*, *W*, *Lb*} — верх,

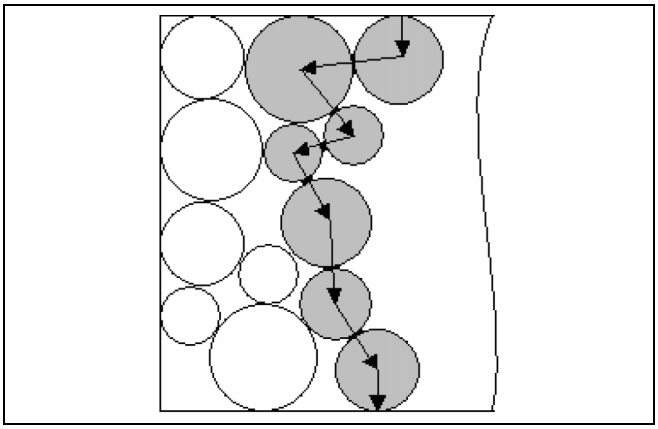

Рис. 7. Схема края размещения

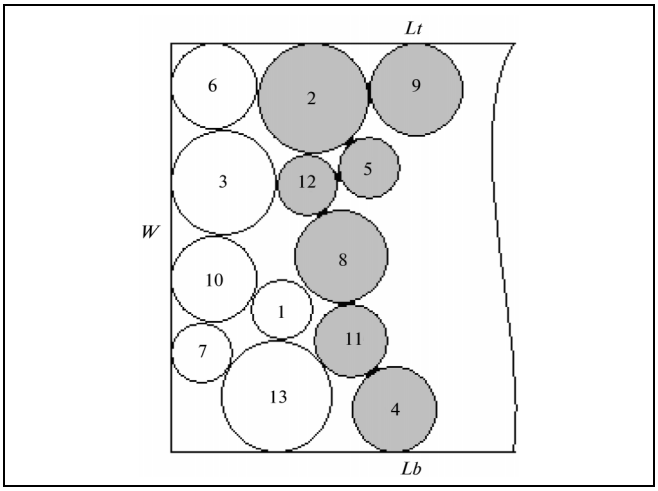

Рис. 8. Формальная запись края размещения {*Lt*, 9, 2, 5, 12, 8, 11, 4, *Lb*}

торец, низ полосы соответственно. При добавлении предметов край видоизменяется. Формальная запись края размещения проиллюстрирована на рис. 8.

Для лучшего описания понятия края упаковки можно провести физическую аналогию. Пусть торец полосы W — дно двумерного стакана, нальем в него воду. Все хотя бы частично намокшие предметы — это край полосы.

При помощи специальной структуры данных краевой декодер хранит информацию о крае текущего размещения и располагает каждый следующий предмет вплотную к краю, минимизируя общую занятую предметами длину полосы.

Для решения обеих задач (прямоугольного и круглого раскроя) предлагается модификация генетического алгоритма, использующая процедуру декодирования как параметр. Сам генетический алгоритм (т. е. схема его работы, смены поколений), структура хромосомы, операторы мутации, скрещивания, селекции не чувствительны к геометрии размещаемых предметов. Например, оператор мутации случайным образом переставляет два номера предметов в хромосоме. Оператор скрещивания двух хромосом порождает двух потомков. Начальный участок дочерней хромосомы совпадает со случайным участком хромосомы одного родителя, а конец состоит из оставшихся предметов, перечисленных в том порядке, в котором они следуют в хромосоме другого из родителей.

Например, оператор скрещивания (для случайных параметров $p = 3$, $q = 4$) для родителей S_{p1} = (1, 2, **3**, **4**, **5**, **6**, 7, 8, 9) и S_{p1} = (4, **7**, **9**, **1**, **2**, 5, 3, **8**, 6) порождает потомка S_{c1} = (**3**, **4**, **5**, **6**, **7**, **9**, **1**, **2**, **8**) и для тех же родителей (для удобства понимания по-другому записанных) S_{p2} = (4, 7, **9**, **1**, **2**, **5**, 3, 8, 6) и S_{p1} = (1, 2, **3**, **4**, 5, **6**, **7**, **8**, 9) — потомка S_{c2} = (**9, 1, 2, 5, 3, 4, 6, 7, 8**). Оператор мутации (для случайных параметров $a = 2$, $b = 7$) исходную особь S_p = (9, **8**, 7, 6, 5, 4, **3**, 2, 1) преобразует в особь вида S_m = (9, **3**, 7, 6, 5, 4, **8**, 2, 1).

Генетический алгоритм, допускающий смену декодеров, и оба декодера были запрограммированы в визуальной среде разработки приложений Delphi. При этом смена декодера в генетическом алгоритме — тоже часть вычислительного эксперимента. Он показал, что оценочная полиномиальная сложность алгоритма и относительно невысокие времена работы программы не зависят от типа задачи. В то время как оценочная сложность блочного декодера пропорциональная квадрату размерности задачи (числу предметов), а краевого — линейно зависит от размерности. Качество размещения прямоугольных предметов получено в среднем не хуже, чем в работах профессора Э.А. Мухачевой Качество размещения круглых предметов оценено специалистами кафедры как удовлетворительное.

Разработанный алгоритм ориентирован на расширение и модификацию. Смена декодера позволяет решать задачу раскроя полубесконечной полосы как на прямоугольные, так и на круглые предметы. Предположительно, если воспользоваться таким эвристическим приемом, как годограф Хэкманна [26], в качестве декодера, то тот же алгоритм возможно использовать для раскроя полубесконечной полосы на предметы нерегулярной формы.

Обе изначально поставленные задачи раскроя предполагают размещение предметов не на полубесконечной полосе, как обсуждалось при формализации задач, а на наборе прямоугольных листов. Разработанный автором алгоритм был модифицирован и для решения задач раскроя в таких, действительно прикладных, а не абстрактных постановках [27].

В реальном производстве, например, металлургическом, все листы имеют одинаковую ширину. Это выгодное обстоятельство дает возможность не рассматривать набор отдельных листов, а условно разбить на них непрерывную полосу. Предложено оставить в математической постановке задачи полубесконечную полосу, но добавить ограничение: предметы не должны пересекать границы между листами [27].

При модификации алгоритма остались неизменными все генетические операторы и сама схема работы генетического алгоритма.

ЗАКЛЮЧЕНИЕ

Как показывает опыт зарубежных и российских исследований [2—24], генетические алгоритмы успешно применяются для решения различных задач оптимизации, где трудно или невозможно воспользоваться другими методами. Среди них — задачи с большим пространством поиска с множеством экстремумов, задачи со сложно формализуемой целевой функцией, с необходимостью многокритериального поиска, поиска приемлемого, рационального, а не оптимального решения в рамках имеющихся ресурсов, многопараметрические задачи, задачи, решаемые в режиме реального времени. Более того, генетические алгоритмы, как видно из представленного в статье исследования, не только позволяют создать конкретные методы, с приемлемой эффективностью решающие отдельные сложные по тем или иным признакам задачи. Структура самого алгоритма дает возможность находить решение других задач того же класса, незначительно и/или определенным образом модифицируя метод, не разрабатывая новой структуры данных, не внедряя нового подхода к решению.

ЛИТЕРАТУРА

1. *Holland J.H.* Adaptation in Natural and Artificial Systems. — The University of Michigan Press, University of Michigan, Ann Arbor, 1975.
2. *Редько В.Г.* Эволюционная кибернетика. — М.: Наука, 2001. — 159 с.
3. *Стецюра Г.Г.* Эволюционные методы в задачах управления, выбора, оптимизации // Приборы и системы управления. — 1998. — № 3. — С. 54—62.
4. *Курейчик В.М.* Генетические алгоритмы. Состояние, проблемы, перспективы // Известия академии наук. Теория и системы управления. — 1999. — № 1. — С. 144—160.
5. *Гудман Э.Д.* Эволюционные вычисления и генетические алгоритмы // Обозрение прикладной и промышленной математики. — 1996. — Т. 3, вып. 5.
6. *Де Янг К.* Эволюционные вычисления: достижения и проблемы // Там же.
7. *Курейчик В.М.* Генетические алгоритмы. Обзор и состояние // Новости искусственного интеллекта. — 1998. — № 3.
8. *Родзин С.И.* Проектирование самотестируемых микросхем с применением метода генетического поиска // Изв. ТРТУ. — 1997. — № 3 (6).
9. *Курейчик В.М., Родзин С.И.* Эволюционные алгоритмы: генетическое программирование // Известия академии наук. Теория и системы управления. — 2002. — № 1. — С. 127—137.
10. *Курейчик В.В.* Эволюционные методы решения оптимизационных задач. Таганрог: Изд-во ТРТУ, 1999.
11. *Куприянов М.С., Матвиенко Н.И.* Генетические алгоритмы и их реализации в системах реального времени // Информационные технологии. — 2001. — № 1. — С. 17—21.
12. *Васильев В.И., Ильясов Б.Г.* Интеллектуальные системы управления с использованием генетических алгоритмов // Приложение к журналу «Информационные технологии». — 2000. — № 12. — 25 с.
13. *Модели* и методы решения задач ортогонального раскроя и упаковки: аналитический обзор и новая технология блочных структур / *Э.А. Мухачева* и др. // Приложение к журналу «Информационные технологии». — 2004. — № 5. — 32 с.
14. *Подлазова А.В.* Генетические алгоритмы в задачах плоского регулярного раскроя // Тр. II Междунар. конф. «Параллельные вычисления и задачи управления (PACO'2004)» / Ин-т пробл. упр. — М., 2004. — С. 284—316.
15. *Подлазова А.В.* Двумерный раскрой и генетические алгоритмы: постановка и алгоритм решения задачи раскроя круглых предметов различного диаметра // Тез. докл. всерос. конф. «Высокопроизводительные вычисления и технологии (ВВТ'2003)» / Ин-т компьютерных исследований. — Ижевск, 2003. — С. 179—184.
16. *Липницкий А.А.* Применение генетических алгоритмов к задаче о размещении прямоугольников // Кибернетика и системный анализ. — 2002. — № 6. — С. 180—184.
17. *Норенков И.П., Косачевский О.Т.* Генетические алгоритмы комбинирования эвристик в задачах дискретной оптимизации // Информационные технологии. — 1999. — № 2. — С. 2—7.
18. *Мухачева А.С., Чиглинцев А.В.* Генетический алгоритм поиска минимума в задачах двумерного гильотинного раскроя // Информационные технологии. — 2001. — № 3. — С. 27—31.
19. *Задачи* двумерной упаковки: развитие генетических алгоритмов на базе смешанных процедур локального поиска оптимального решения / *Э.А. Мухачева* и др. // Приложение к журналу «Информационные технологии». — 2001. — № 9. — 24 с.
20. *Мухачева Э.А., Мухачева А.С., Чиглинцев А.В.* Генетический алгоритм блочной структуры в задачах двумерной упаковки // Информационные технологии. — 1999. — № 11. — С. 13—18.
21. *Росс Клемент* Генетические алгоритмы: почему они работают? Когда их применять? // Компьютерра. — 1999. — № 11. — С. 20—23.
22. *Hopper H., Turton B.C.H.* A review of the application of meta-heuristic algorithms to 2D strip packing problems // Artificial Intelligence Review. — 2001. — N 16. — P. 257—285.
23. *Емельянов В.В., Курейчик В.М., Курейчик В.В.* Теория и практика эволюционного моделирования. — М.: Физматлит, 2003. — 432 с.
24. *Рутковская Д., Пилиньский М., Рутковский Л.* Нейронные сети, генетические алгоритмы и нечеткие системы — М.: Горячая линия — Телеком, 2004. — 452 с.
25. *Канторович Л.В., Залгаллер В.А.* Рациональный раскрой промышленных материалов. — Новосибирск: Наука, 1971. — 300 с.
26. *Верхотуров М.А.* Задача нерегулярного раскроя плоских геометрических объектов: моделирование и расчет рационального раскроя // Информационные технологии. — 2000. — № 5. — С. 37—42.
27. *Подлазова А.В.* Разработка метода эффективного решения задач плоского раскроя с использованием генетических алгоритмов: Автореф. ... дис. канд. техн. наук. — М.: МИСиС, 2004. — 22 с.

e-mail: podlazova@yandex.ru

Статья представлена к публикации членом редколлегии А.С. Рыковым. ❑