ПРИКЛАДНАЯ МАТЕМАТИКА И МЕХАНИКА

Вестник Сыктывкарского университета.
Серия 1: Математика. Механика. Информатика.
Выпуск 3 (24). 2017

УДК 519.8

МЕТОДЫ ПРЕДСТАВЛЕНИЯ ФИГУР ОБЩЕГО ВИДА ДЛЯ ЗАДАЧИ ДВУМЕРНОГО РАСКРОЯ

В. А. Мельников

В статье описан способ представления фигур в виде растровых матриц. Также приведён алгоритм дополнительной обработки данных в матрицах для снижения сложности дальнейших вычислений. В последней части даны принципы позиционирования фигур на заготовке.

Ключевые слова: фигуры общего вида, раскрой, двумерное пространство, принцип левого нижнего угла.

1. Введение в задачу раскроя материала

Экономия материала представляет собой сложную и важную проблему, с которой часто приходится встречаться на различных производствах, при резке различных материалов на листы металла, стекла или дерева, трубы, профильный прокат, изделия сложной формы. Для её решения необходимо отыскать такой способ расстановки заготовок на материале, из которого они вырезаются, чтобы отходы после резки были минимальны. Оптимизация использования материалов позволяет достичь большой экономии денежных средств.

На самом деле, задача раскроя является NP-полной даже для прямоугольников. Для фигур неправильной формы геометрическая сложность увеличивает количество совершаемых вычислений, поэтому применяются различные эвристические методы решения задачи.

© Мельников В. А., 2017.

2. Основные характеристики задач раскроя

Прежде чем приступать к рассмотрению алгоритмов решения задачи раскроя, следует рассмотреть характеристики, влияющие на то, как будет выглядеть итоговый алгоритм решения. В статье [1] Harald Dyckhoff приводит достаточно полное описание характеристик задач раскроя.

2.1. Пространственные характеристики

Основная характеристика раскроя — размерность пространства, для которого поставлена задача:

- раскрой в одномерном пространстве;
- раскрой в двумерном пространстве;
- раскрой в трёхмерном пространстве.

Например, загрузка поддонов является задачей в двумерном пространстве. В отличие от задач в двух и более измерениях задача в одномерном пространстве имеет явное решение. Достаточно подробно данная задача описывается в книге Канторовича – Залгаллера [2]. Так же в данной книге можно найти методы решения задач двумерного раскроя для прямоугольных заготовок.

2.2. Количественные характеристики

Другая важная характеристика — количественная. В задаче раскроя необходимо некоторым образом измерять количественные и качественные характеристики фигур, например площадь, длину и ширину фигур. Или количество уже расположенных фигур. Тут можно рассмотреть два варианта:

- дискретное измерение с помощью, например, натуральных и целых чисел;
- дробное измерение на основе вещественных чисел.

Первый вариант позволяет нам подсчитывать количество изделий, уже расположенных на материале, а с помощью второго можно измерять различные характеристики фигур, такие как площадь, длина и ширина.

2.3. Геометрические характеристики

Немалую роль играют в раскрое сами фигуры, которые необходимо расположить на плоскости. В пространстве фигуры однозначно определяются с помощью следующих свойств:

- формы;
- размера;
- ориентации.

2.4. Характеристики по ограничениям на результат

По ограничениям на результат можно выделить четыре основные группы:

- минимальное расстояние между объектами;
- ориентация фигур относительно друг друга;
- ограничение на количество фигур;
- ограничение на количество совершаемых «резов».

Автор статьи [1] выделяет ещё несколько групп по различным признакам, но указанные выше являются основными.

3. Постановка задачи

Прежде чем переходить к алгоритмам, применяемым для решения задачи раскроя, следует рассмотреть постановку задачи. Задача в двумерном пространстве ставится по аналогии с задачей раскроя в одномерном пространстве [3], хотя однозначно с помощью математических соотношений описать её нельзя.

Первоначально имеем множество фигур F и $|F| = n$, плоскость с шириной w и высотой h. Необходимо отыскать такое упорядоченное множество F', что $F' \subset F$ и $|F'| \to \max$, при условии что все фигуры из F' будут без пересечений размещены на плоскости с заданными параметрами.

4. Методы представления фигур для раскроя

Самым видимым атрибутом задач раскроя и тем, с чем сразу сталкиваются исследователи в данной области, является геометрическое представление фигур. Решение о том, как представлять фигуры, оказывает решающее значение на дальнейшую разработку системы.

4.1. Методы перемещения фигур

Прежде чем перейти к рассмотрению методов представления фигур, следует ответить на важный вопрос, обычно опускаемый в литературе, связанной с раскроем: «Каким образом перемещать фигуры относительно друг друга?».

Предположим, что мы уже некоторым образом расположили первую фигуру. Она располагается всегда в левом нижнем углу, как на рис. 1. Для удобства границы контейнера изображать не будем. Условимся, что вторая фигура в конечном расположении будет закрашена серым цветом.

Самый первый метод, который будет интуитивно понятен всем, — «лестничный» или же «тетрисный». Его суть заключается в том, что

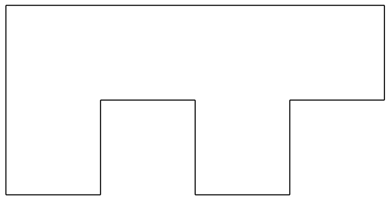

Рис. 1. Расположение первой фигуры

мы двигаем фигуру вниз до первого столкновения с другой, потом также влево, потом опять вниз и так далее, пока фигура не перестанет смещаться. Рассмотрим на примере, как «тетрисный» способ расположит следующую фигуру. На рис. 2 видно, что фигура перемещается слева направо с некоторым шагом по оси x и в итоге находит углубление внутри другой фигуры.

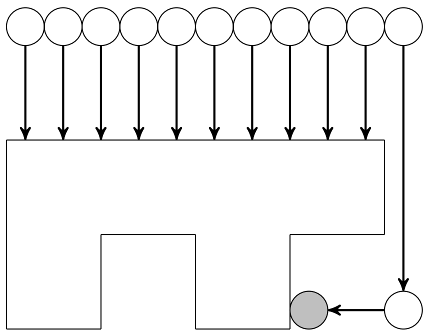

Рис. 2. Расположение второй фигуры «тетрисным» способом

Более сложный метод — «метод сквозного прохода». Его основная идея заключается в том, что фигуру просто перемещают сквозь остальные и ищут подходящее ей место. Как видно на рис. 3, данный метод нашёл закрытую полость, до которой предыдущий способ дойти не смог.

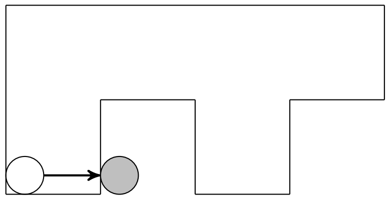

Рис. 3. Расположение второй фигуры способом «сквозного прохода»

Самый сложный метод — движение вдоль контура. Под контуром в данном случае подразумевается обновляемый контейнер. Изначально имеется некоторый контур пустого контейнера, затем в него добавляются одна за другой фигуры, и контур после расположения каждой из них изменяется. Таким образом, фигура движется вдоль контура контейнера, при нахождении подходящего места необходимо обновить контур с учетом расположения новой фигуры. На данный момент этот метод будет оставлен без особого внимания. Для его реализации необходимо создать цепь (растровую или же векторную), вдоль которой будет перемещаться фигура. Большую сложность представляет в данном случае вопрос о выборе точки, относительно которой идёт движение.

4.2. Представление фигур в виде многоугольников

Представление фигур в виде многоугольников даёт хорошую точность аппроксимации. В таком представлении объем информации пропорционален числу вершин и не зависит от размера фигуры. Полигональное представление является первичным для фигур, а на его основе уже можно построить растровое представление, которое будет описано ниже.

Полигональный метод хоть и даёт высокую точность представления, но имеет очень высокую вычислительную сложность — $\mathrm{O}(e^n)$, где n — число вершин фигуры.

Для того чтобы проверить, нет ли пересечений между какими-либо фигурами, нужно выполнить следующий набор тестов [6]:

1. Проверить, пересекаются ли описывающие прямоугольники фигур, если нет, то и фигуры не пересекаются, иначе перейти к следующему тесту.
2. Для каждой пары рёбер проверить, пересекаются ли их описывающие прямоугольники.
3. Проверить, пересекаются ли рёбра.
4. Проверить, лежат ли какие-либо вершины одного полигона внутри другого.

Проверка на пересечения текущей фигуры с ранее расположенными может выполняться различными способами. Первый вариант — через уравнение прямой с угловым коэффициентом. Этот метод для данной задачи будет излишним, ведь, кроме проверки на пересечение двух отрезков, будут найдены точка пересечения и угловой коэффициент k данной прямой. Также придётся проверять специальные случаи, когда пря-

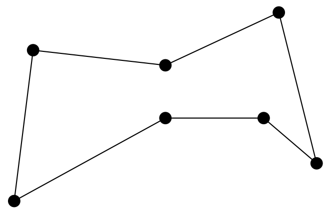

Рис. 4. Представление фигуры при помощи многоугольника

мые параллельны, когда они направлены вертикально вверх. Лучше использовать метод проверки на основе псевдоскалярного произведения, ведь тогда сильно снизится вычислительная нагрузка. Данный метод подробно рассматривается в задачах вычислительной геометрии [4].

Для перемещения фигуры подходят только алгоритмы «тетрисного» движения. Применять «сквозное движение» мы не можем, так как тогда придется постоянно проверять, не попала ли она внутрь другой, что в случае многоугольников сделать достаточно сложно. Для этого необходимо применять метод «трассировки луча» или приближённо считать комплексный интеграл, пользуясь интегральной формулой Коши [5].

Алгоритмы с представлением фигур в виде полигонов хорошо подходят для прямоугольников и несложных многоугольников с числом вершин до сотни. Для более сложных сильно возрастает время вычисления пересечений.

Проблема данного метода состоит в сложности обработки контура. Построить эквидистантный (равноудалённый во всех точках от исходного) контур, чтобы задать зазор — не такая уж лёгкая задача. Отделить внешний контур от внутренних тоже сложно, ведь контур может состоять из нескольких раздельных кривых, пусть и образующих в сумме одну замкнутую кривую.

4.3. Представление фигур в виде растровых матриц

Растровый метод позволяет упростить геометрическую сложность фигуры и без дополнительных оптимизаций снизить сложность вычислений до $O(n^2)$.

Растровые методы предлагают разделить непрерывный раскройный лист на дискретные части, упрощая сложную геометрическую информацию до представления матрицей. Под матрицей будем понимать некоторое представление растра, в котором отмечены занятые и свободные места. Существуют различные методы представления.

Самый простой метод представления — это 1 для занятого деталью места и 0 для свободного. Раскраиваемый материал в данном случае представляется аналогично. На рис. 5 можно увидеть первичный вариант растрового представления [6]. В данном случае занятые области отмечены серым цветом.

Растровый способ не является первичным, он строится из фигуры, представленной многоугольником. Поэтому нельзя сразу говорить, какие места заняты, а какие нет, ведь на входе имеется простое наложение многоугольника на растровую матрицу, что можно увидеть на рис. 6. Рассмотрим возможный алгоритм закраски занятых областей:

1. На первом шаге имеется исходный многоугольник и пустой растр, как показано на рис. 6. Выберем любое его ребро.

2. Так как каждое ребро задается координатами (x_1, y_1) и (x_2, y_2), то на втором шаге можно получить уравнение прямой с угловым коэффициентом, которая содержит данный отрезок. Для определённости будем считать, что $y_1 \leq y_2$.

3. Зададим множество $A = \{a_0, a_1, a_2, ..., a_{n-1}, a_n\}$, где $a_0 = y_1$, а $a_n = y_2$. Остальные элементы являются целыми числами из интервала (y_1, y_2).

4. Теперь для каждой пары чисел $\{a_i, a_{i+1}\}$, $i = 0 : n-1$ из имеющегося уравнения прямой $y = kx + b$ вычислим x_i и x_{i+1}.

5. Вычислив все необходимые координаты, можно отметить занятыми (на изображениях закрашены серым цветом) все клетки на отрезке, заданном координатами (x_i, a_i), (x_{i+1}, a_i).

Алгоритм рассматривает переход от отрезка к растру. Для перехода от многоугольника к растру достаточно повторить описанный алгоритм для всех его ребёр. В итоге получится то, что изображено на рис. 5.

Операция проверки на пересечения с другими фигурами при текущем расположении в координате (x, y) фигуры с шириной, равной w, и высотой h не требует никаких сложных действий. Производится простое суммирование по следующей формуле:

$$S = \sum_{i=1}^{h} \sum_{j=1}^{w} p_{i+y,j+x} f_{i,j}. \qquad (1)$$

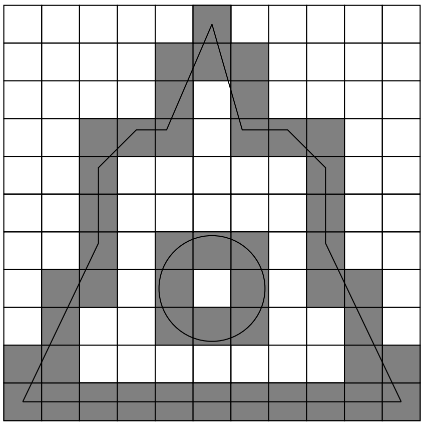

Рис. 5. Первичная растровая матрица

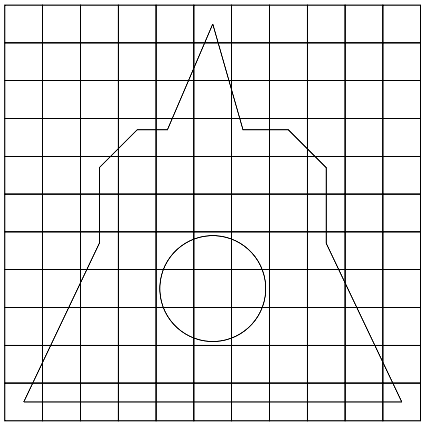

Рис. 6. Наложение многоугольника на растровую матрицу

В данном случае, $p_{i+y,j+x}$ — значение в матрице раскройной плоскости со смещением на координату (x, y), а $f_{i,j}$ — значение в матрице фигуры. В случае если значение $S > 0$, то имеется пересечение с некоторой фигурой, иначе его нет.

Как видно на рис. 5, не все области внутри фигуры отмечены как занятые. В таком случае некоторые фигуры могут попасть внутрь других, что будет являться ошибкой. Данная проблема может быть решена различными способами. Самый простой способ — «залить» всё, что находится внутри внешнего контура, отбрасывая пустоты внутри фигуры. Такой метод принесёт большие потери материала.

Рассмотрим метод, позволяющий учитывать пустые места внутри фигур. Для начала необходимо отделить внешний контур фигуры:

1. Примем за текущую самую левую занятую точку с наименьшей ординатой.

2. Отметим текущую точку как точку контура (на изображениях будет отмечаться диагональной штриховкой).

3. Рассмотрим всех соседей текущей точки. Если есть соседняя точка, отмеченная цифрой 1, переходим в данную точку и возвращаемся на второй шаг. Если текущая точка — исходная, то закончить алгоритм, иначе повторить текущий шаг.

Результат работы алгоритма можно увидеть на рис. 7.

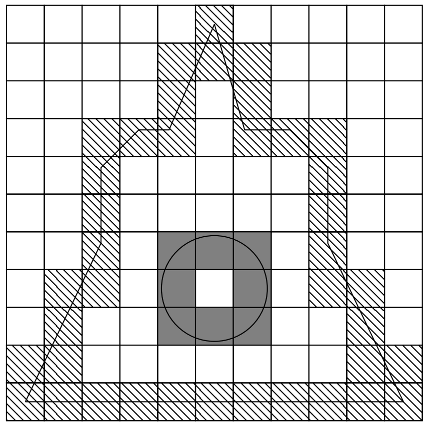

Рис. 7. Фигура с выделенным контуром

После отделения внешнего контура необходимо отметить занятым всё место между внешним контуром и контурами второго уровня. Это можно сделать следующим образом:

1. Первым шагом необходимо сохранить точки контура в отдельный массив и отметить их как свободные (на изображениях не закрашены цветом) на растровой матрице.

2. Далее выполняется «заливка». В процессе «заливки» свободные клетки отмечаются как временно занятые (на изображениях будет изображаться перекрёстной штриховкой). Результат первых двух шагов приведён на рис. 8.

3. Третьим шагом инвертируем «заливку». Для этого необходимо все временно занятые клетки отметить как свободные. Свободные клетки необходимо отметить как временно занятые. Занятые клетки не затрагиваются. После этого можно вернуть на растровой матрице контур фигуры. Результат данного шага приведён на рис. 9.

4. После этого опять выполняется «заливка». Результат четвёртого шага приведён на рис. 10.

5. Последним шагом опять выполняется инверсия «заливки» по тем же правилам, что и на третьем шаге, только свободные клетки теперь отмечаются как занятые. Результат работы алгоритма приведён на рис. 11.

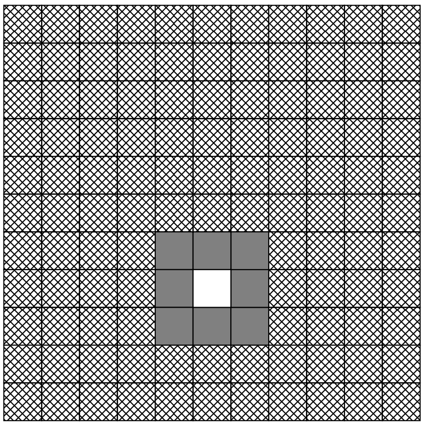

Рис. 8. Результат первых двух шагов алгоритма «заливки» пустых мест

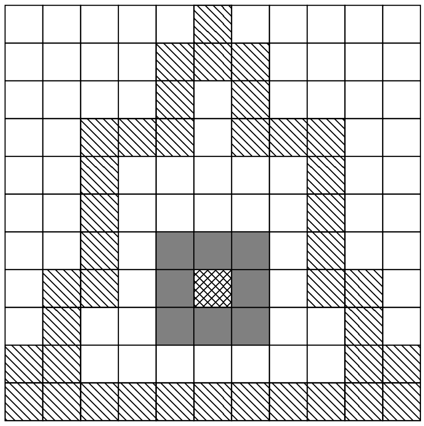

Рис. 9. Результат третьего шага алгоритма «заливки» пустых мест

4.4. Условия корректности данных

Для корректности работы описанных выше алгоритмов и некоторых дальнейших выкладок необходимо ввести условия корректности.

Фигура является корректной, если выполняются следующие условия:

1. Фигура имеет замкнутый внешний контур.

2. Уровень вложенности контуров не более первого. Это означает, что контуры могут вкладываться только во внешний контур, то есть если контур является вложенным во внешний, то в него уже не может быть вложен контур.

3. Вложенные контуры не должны иметь пересечений друг с другом.

4. Ни один из контуров не должен иметь самопересечений.

Несоблюдение приведённых выше условий может привести к непредсказуемым результатам работы алгоритма.

4.5. Уменьшение сложности до линейной

Для корректности дальнейших действий введём условие, что движение фигур по плоскости материала происходит строго построчно. Из данного утверждения следует тот факт, что если фигура пересекает занятое другой фигурой место, то хотя бы в одной из точек пересечение будет лежать на контуре текущей фигуры.

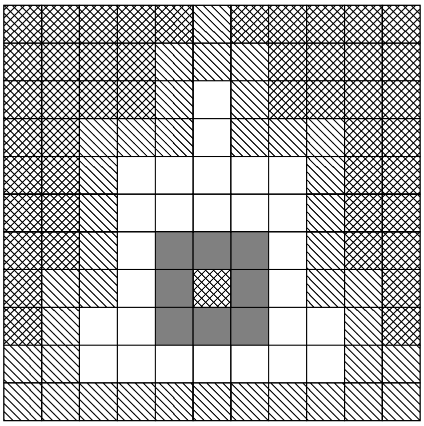

Рис. 10. Результат четвёртого шага алгоритма «заливки» пустых мест

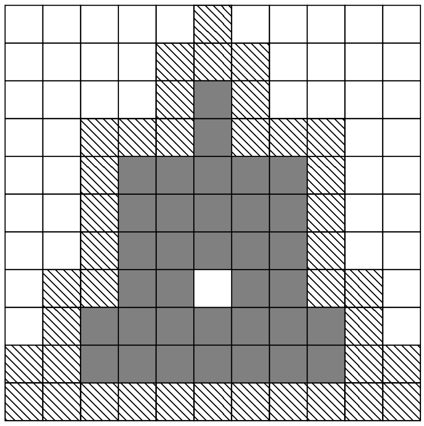

Рис. 11. Итоговый результат работы алгоритма «заливки» пустых мест

Тогда можно снизить сложность поиска пересечений до $\mathrm{O}(n)$, где n — число точек, входящих в контур. Такой способ поиска пересечений является суммированием по следующей формуле:

$$S = \sum_{(i,j)\in\Gamma} p_{i+y,j+x}. \tag{2}$$

Данная формула аналогична формуле (1), но суммирование проходит только по тем точкам (j, i), которые входят во внешний контур Γ.

5. Принципы позиционирования фигур

Определившись с представлением фигур и методом их перемещения, разработчику необходимо решить, каким образом производить позиционирование. Первое, с чего стоит начать, — это поворот фигур. Некоторые авторы предлагают применять изменение угла методом дихотомии, что по сути является бинарным поиском. Но тут возникает большая проблема. Бинарный поиск работает на отсортированном множестве, а контур фигуры таковым не является, поэтому поворот следует выполнять просто с дискретным шагом изменения угла.

После того как фигура уже лежит где-то на плоскости, её положение надо оценить. Очень часто применяют принцип левого нижнего угла (ЛН-принцип). Так как фигуры имеют сложный контур, можно предложить следующее улучшение данного метода. Пользуясь тем, что точки, входящие в контур, известны, можно найти его центр масс по следующей формуле:

$$p_g = \sum_{(i,j)\in\Gamma} \frac{p_{i,j}}{n}, \tag{3}$$

где $p_{i,j}$ — это точка, входящая в контур Г. Теперь можно не только пытаться расположить фигуру максимально влево и вниз, но и в случае, если два положения имеют одинаковую высоту, выбрать то, у которого центр масс фигуры ниже.

Рассмотрим на примере, как таким способом можно расположить некоторый пятиугольник. На рис. 12 видно, что пятиугольник расположен центром масс вниз, сверху та часть, которая оставляет больше свободного места для расположения оставшихся фигур.

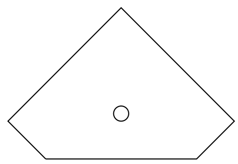

Рис. 12. Модифицированный ЛН-принцип

Можно придумать множество дальнейших модификаций данного метода, например, сначала оценивать описывающий прямоугольник множества фигур и выбирать тот вариант расположения, при котором он меньше.

6. Заключение

В данной работе приведён обзор наиболее популярных способов представления фигур. Для растровых матриц были приведены авторские модификации и дополнения к представлению фигур с целью снизить вычислительную сложность. Важным преимуществом использования растровых матриц в задаче раскроя является простота реализации метода. Принимая за реализацию векторных первичных данных формат SVG, получаем, что 1 мм равен $3,543307$ пкс [7], это позволяет говорить о достаточной точности для технических процессов со строгим допуском при использовании зазора между фигурами. Последняя часть работы предлагает метод позиционирования фигур с учётом их геометрических особенностей.

Список литературы

1. **Dyckhoff H.** A typology of cutting and packing problems // *European Journal of Operational Research. № 44. Pp. 150—152.*

2. **Залгаллер В. А., Канторович Л. В.** Рациональный раскрой промышленных материалов. Новосибирск: Наука, 1971. 300 с.

3. **Никитенков В. Л., Холопов А. А.** Задачи линейного программирования и методы их решения. Сыктывкар: Изд-во Сыктывкарского университета, 2008. 143 с.

4. **Прасолов В. В.** Задачи по планиметрии. 4-е изд., доп. М.: МЦНМО, 2001. 584 с.

5. **Шабат Б. В.** Введение в комплексный анализ. М.: Наука, 1969. 91 с.

6. **Benell A. J., Olivera F. J.** The geometry of nesting problems: A tutorial // *European Journal of Operational Research. 2008. № 184. Pp. 399—402.*

7. Coordinate Systems, Transformations and Units // https://www.w3.org: W3C. 6 мая 2017. URL: https://www.w3.org/TR/SVG/coords.html.W3C (дата обращения: 05.10.2017)

Summary

Melnikov V. A. Methods for representing figures of general kind for a two-dimensional cutting problem

This article introduces representation of figures as rastr matrixes. Also the algorithm of additional data processing for reducing future computations difficulty is given. The last part describes principles of positioning figures on the workpiece.
Keywords: figures of general kind, cutting, two-dimensional space, bottom-left principle.

References

1. **Dyckhoff H.** A typology of cutting and packing problems, *European Journal of Operational Research*, № 44, pp. 150—152.

2. **Zalgaller V. A., Kantorovich L. V.** *Ratsionalnyi raskroi promyshlennyh materialov* (Rational cutting of industrial materials), Novosibirks: Nauka, 1971, 300 p.

3. **Nikitenkov V. L., Holopov A. A.** *Zadachi lineynogo programmiriovaniya i metody ih resesheniya* (Linear programming problems and methods for their solution), Syktyvkar: Izdatelstvo Syktyvkarskogo universiteta, 2008, 143 p.

4. **Prasolov V. V.** *Zadachi po planimetrii* (Planning problems), 4th ed., dopolnennoe, M.: MCNMO, 2001, 584 p.

5. **Shabat B. V.** *Vvedenie v kompleksnyi analiz* (Introduction to complex analysis), M.: Nauka, 1969, 91 p.

6. **Benell A. J., Olivera F. J.** The geometry of nesting problems: A tutorial, *European Journal of Operational Research*, 2008, № 184, pp. 399—402.

7. Coordinate Systems, Transformations and Units // https://www.w3.org: W3C. 6 мая 2017. URL: https://www.w3.org/TR/SVG/coords.html.W3C (date of the application: 05.10.2017)

Для цитирования: Мельников В. А. Методы представления фигур общего вида для задачи двумерного раскроя // *Вестник Сыктывкарского университета. Сер. 1: Математика. Механика. Информатика. 2017. Вып. 3 (24). С. 11–24.*

For citation: Melnikov V. A. Methods for representing figures of general kind for a two-dimensional cutting problem, *Bulletin of Syktyvkar University. Series 1: Mathematics. Mechanics. Informatics*, 2017, №3 (24), pp. 11–24.

СГУ им. Питирима Сорокина *Поступила 05.10.2017*