# Один способ нахождения кратчайшей ломаной с вершинами на контурах в плоскости

# Введение

При разработке управляющих программ для машин фигурной листовой резки с ЧПУ возникает ряд оптимизационных задач, одной из которых является задача минимизации холостого хода инструмента, которая может быть сведена к задаче нахождения кратчайшей ломаной с вершинами на плоских контурах. Контуры интерпретируются как границы плоских деталей. Размещение контуров на плоскости определяется решением задачи фигурного раскроя (задача «нестинга»). Обе задачи, в общем случае, является NP-трудными. В свою очередь, задача нахождения холостого хода инструмента является подзадачей другой оптимизационной задачи – задачи оптимизации маршрута резки плоских деталей. В силу NP трудности, её точное решение не может быть получено для реально возникающих в производстве задач (для сотен деталей) за разумное время, поэтому для её решения (так же, как и для задачи раскроя) применяются разнообразные эвристики, дающие решения приемлемого качества. В то же время вопрос оценки качества полученных решений в сравнении с оптимальным решением как правило остаётся нерешённым и представляет значительный научный интерес.

## Классификация задач резки

Общая задача маршрутизации резки 2-мерных плоских деталей как правило даже не ставится ввиду её большой сложности. В литературе можно найти много вариантов её частных случаев, среди которых наиболее популярны (см. рис. 1):

* Задача непрерывной резки (CCP, Continuous Cutting Problem): каждый замкнутый контур (ограничивающий деталь) вырезается целиком одним движением резака, но резка может начаться с любой точки (и в ней же завершиться)
* Обобщённая задача коммивояжёра (GTSP, Generalized Traveling Salesman Problem): резка может начаться только в одной из заранее заданных точек на контуре, контур должен быть вырезан целиком
* Задача резки с конечными точками (ECP, Endpoint Cutting Problem): резка может начаться только в одной из заранее заданных точек на контуре, причём контур может быть вырезан в несколько подходов, по частям.
* Задача непрерывной сегментной резки (SCCP, Segment Continuous Cutting Problem): вводится понятие сегмента резки (представляющее собой обобщение контура; может быть как частью контура, так и объединением нескольких контуров или их частей), сегмент вырезается целиком, таким образом .
* Задача обобщённой сегментной резки (GSCCP, Generalized Segment Continuous Cutting Problem): сегментная резка (SCCP), но выбор сегментов не фиксирован заранее, а сам подлежит оптимизации
* Задача прерывистой резки (ICP, Intermittent Cutting Problem) – наиболее общая описанная в литературе формулировка задачи резки, когда контура могут резаться по частям, в несколько подходов и резка может начаться в любой точке контура.

GSCCP

GSCCP

TSP

GTSP

CCP

ECP

SCCP

ICP

GSCCP

Конечное множество точек врезки

Фиксированные точки врезки

Непрерывное множество точек врезки

Рис 1. Классификация задач маршрутизации резки

Задача резки в общем случае представляет собой сложную задачу непрерывно-дискретной оптимизации с довольно сложной целевой функцией. На практике она очень часто сводится к задаче дискретной оптимизации за счёт дискретизации контуров, подлежащих резке с некоторым шагом , то есть сводится к задаче ECP или её частному случаю – GTSP.

При этом, однако, общая погрешность длины пути резки или холостого хода достигает , где – количество контуров. Чтобы гарантировать точность результата , требуется выбирать малое , таким образом общее количество точек быстро растёт (как ) и полный перебор становится экспоненциальным (). Такие задачи тем не менее могут успешно решаться, например, методом динамического программирования, для небольших даже точно.

В данной статье рассматривается задача маршрутизации резки без использования дискретизации (CCP).

## Технологические ограничения

Необходимость исполнения полученного маршрута на машине листовой резки с ЧПУ накладывает на решение математической задачи резки (например, GTSP) целый ряд ограничений.

Наиболее хорошо описано в литературе так называемое «ограничение предшествования» (precedence constraint), вызванное тем, что после вырезания замкнутого контура, его внутренность как правило ничем не удерживается и может сдвигаться, поворачиваться и даже падать. По этой причине внутренние контура деталей должны вырезаться прежде, чем содержащие их внешние контура, а детали, расположенные в отверстиях больших деталей – ещё раньше. В случае задач ECP и ICP эти формулировки ещё усложняются ввиду того, что контуры деталей могут вырезаться не за один раз.

Другим интересным ограничением (не рассматриваемым в данной статье), вытекающим из той же технологической особенности, является необходимость учёта возможности столкновения режущей головки с ранее вырезанными деталями.

Наконец, в большинстве технологий резки требуется, чтобы резка осуществлялась не строго по контуру, а с некоторым отступом. Этот сдвиг может выполняться как в процессе решения задачи маршрутизации, так и после – на этапе генерации управляющей программы для машины резки с ЧПУ или даже самим станком непосредственно в процессе резки. Кроме того, точка врезки (точка включения инструмента) как правило должна быть расположена на ещё большем расстоянии от контура во избежание повреждения детали. Однако, в данной статье это требование не рассматривается. Таким образом, далее везде считается, что инструмент движется точно по контуру детали и точка врезки расположена прямо на контуре (как и точка выключения инструмента).

# Задача непрерывной резки

Рассмотрим эвклидову плоскость и на ней – область , ограниченную замкнутым контуром (как правило – прямоугольник), которая представляет собой модель листового материала, подлежащего резке. Пусть внутри задано попарно непересекающихся плоских контуров , ограничивающих деталей . Деталь может быть ограничена как одним контуром, так и несколькими (внешним и внутренними отверстиями), так что в общем случае .

Контуры могут иметь произвольную форму, но мы будем рассматривать только случай, когда они состоят из (конечного числа) отрезков прямых и дуг окружностей, что определяется существующим технологическим оборудованием. В частном случае, когда контуры состоят только из отрезков прямых, задача непрерывной резки сводится к одному из вариантов задачи обхода многоугольников (Touring Polygon Problem, TPP).

Далее, в области (как правило, на её границе) задаются две точки, обозначим их (на практике как правило ), представляющие собой начало и конец маршрута резки.

Задача непрерывной резки заключается в поиске:

1. точек врезки
2. Порядка обхода контуров , то есть перестановки элементов

Результатом решения задачи будет маршрут . Целевая функция в данном случае сильно упрощается по сравнению с общей задачей резки и сводится к минимизации длины холостого хода

Где для простоты мы вводим обозначения .

Кроме того, мы будем решать задачу оптимизации с дополнительным ограничением, так называемым «ограничением предшествования» (precedence constraint). Хотя контуры не пересекаются, они могут быть вложены друг в друга, то есть , где обозначает 2-мерную фигуру, ограниченную контуром (в более привычных обозначениях ). В общей задаче резки это может быть вызвано двумя разными обстоятельствами (отверстия в деталях и размещение меньших деталей в отверстиях больших для экономии материала), но в данной задаче эти варианты обрабатываются одинаково.

Если один контур расположен внутри другого, то вложенный контур должен быть вырезан (посещён) ранее внешнего: в перестановке. Таким образом, не все перестановки контуров оказываются допустимы.

# Общий алгоритм решения

Предлагаемый алгоритм решения состоит из нескольких этапов, естественно связанных с природой решаемой задачи.

## Удаление внешних контуров

Для автоматического соблюдения ограничений предшествования на первом этапе удаляются все контуры, содержащие вложенные, то есть остаются только контура, удовлетворяющие условию:

Это в общем случае приводит к уменьшению размера задачи (в некоторых случаях – значительному), и тем самым сокращает время расчётов на втором и особенно третьем этапе.

## Непрерывная оптимизация

Mi-1

Mi+1

Ci-1

Ci+1

Ci

Mi

M’i

Mi-1

Mi+1

Ci-1

Ci+1

Ci

Mi

M’i

M\*i-1

Рис. 2. Выбор оптимальной точки врезки

На этом этапе мы предполагаем последовательность обхода заданной и ищем координаты точек врезки в каждый контур , минимизирующих суммарную длину холостого хода. Для этого выбираются начальные положения точек (например, случайным образом) и положение каждой точки изменяется в предположении, что все остальные неподвижны: . Большая часть слагаемых в целевой функции константны, поэтому она упрощается до

Простейший геометрический анализ показывает, что неформально говоря, если точки расположены по разные стороны сегмента контура , то оптимальная позиция точки врезки – на пересечении с отрезком: (если такое пересечение существует; в противном случае решением будет один из концов сегмента). В случае же, если точки расположены по одну сторону сегмента, то решение также легко находится с применением принципа Ферма, порождающем знаменитое правило «угол падения равен углу отражения» (либо снова один из концов сегмента), см. рис. 2.

Общая схема оптимизации на данном этапе, таким образом выглядит так:

* Выбираем произвольные начальные положения точек врезки .
* находим оптимальную позицию как описано выше за константное время .
* Повторяем предыдущий шаг, пока позиции всех точек не сойдутся (с некоторой заранее заданной точностью .

На практике весь процесс хорошо сходится за время и поэтому многократно используется как подпрограмма на следующем шаге.

## Дискретная оптимизация

Самый вычислительно сложный этап заключается в поиске перестановки , минимизирующей длину холостого хода , то есть фактически решение задачи коммивояжера (Travelling Salesman Problem, TSP) с функцией расстояния, вычисляемой при помощи непрерывной оптимизации, как описано на предыдущем шаге.

Для поиска решения используется эвристический метод переменных окрестностей (Variable Neighborhood Search, VNS). Схема его применения:

1. Начальная перестановка выбирается случайным образом
2. Пока
   1. Из окрестности  выбирается перестановка , минимизирующая
   2. Если
   3. Иначе
3. Конец

На шаге 3.1 многократно применяется шаг непрерывной оптимизации:

Для построения окрестностей разного размера применяются различные приёмы, например:

* Все возможные парные перестановки (фактически это окрестность размера 1 в смысле транспозиционной метрики)
* Циклические перестановки 3 контуров. Все такие перестановки потребовали бы времени , поэтому выбираются только такие варианты, в которых переставляемые контуры удалены в исходной перестановке не более, чем на заранее заданное расстояние (параметр алгоритма).
* Аналогичным образом применяются циклические перестановки 4 контуров, находящихся в пределах некоторого расстояния друг от друга в исходной перестановке .
* Выбирается последовательный блок контуров произвольной длины и осуществляется их циклическая перестановка.
* Перестановка контуров в последовательном блоке произвольной длины «задом наперёд»
* Обмен двух последовательных (но не соседних) блоков контуров
* Циклический сдвиг контуров между несколькими последовательными блоками одинаковой длины
* И ещё порядка десяти различных рецептов построения «близких» перестановок

Если размер окрестности, генерируемой некоторым приёмом, оказывается слишком велик, он может быть легко ограничен введением дополнительного параметра, аналогично тому, как это сделано для тройных и четверных перестановок.

Кроме того, метод переменных окрестностей имеет некоторые вариации, уменьшающие перебор на шаге 3.1, такие как «Первый подходящий» или метод Монте-Карло, однако их влияние на качество и скорость решения задачи непрерывной резки нуждается в дальнейшем исследовании.

## Восстановление удалённых контуров

После того как маршрут инструмента для обхода контуров, не содержащих внутри себя других, построен, как описано выше, достроим его до полного маршрута, посещающего все исходные контуры, причём так, чтобы соблюдалось ограничение предшествования.

Обратим внимание, что полученный на предыдущем шаге маршрут

пересекает все исходные контуры , потому что оставленные на первом этапе контуры он посещает по построению на шагах 2 и 3, а все внешние контура он пересекает согласно теореме Жордана, потому что начальная и конечная точки и лежат снаружи всех контуров .

Таким образом для каждого (внешнего) контура , ещё не включённого в маршрут, мы находим все точки пересечения с ним маршрута, построенного на предыдущем шаге и если таких точек оказывается несколько (как правило), то выбираем из них самую последнюю (посещаемую маршрутом позже всех остальных), см. Рис. 3. После добавления полученных таким образом точек врезки мы получаем уже полный маршрут, который, с одной стороны, посещает все исходные контуры, и при этом внешние контуры всегда посещаются позже, чем содержащиеся в них внутренние. Длина маршрута, очевидно, при этом не изменяется.

Внешний контур

Mi

Mi-1

Mi-2

Mi+1

Mi+2

Добавочная точка врезки

Рис 3. Построение дополнительных точек врезки

Легко понять, что построенный таким образом маршрут для полной задачи также является оптимальным. Действительно, если бы существовал более короткий маршрут, обходящий все контуры, мы могли бы простым удалением из него точек врезки, лежащих на внешних контурах получить маршрут обхода только внутренних контуров, имеющий ту же, то есть меньшую длину. Тем самым мы бы построили более короткий маршрут для подзадачи без ограничений предшествования, что по предположению невозможно.

Таким образом, мы выполняем ограничение предшествования почти автоматически за линейное время .

# Оптимальность решения задачи непрерывной оптимизации

С практической точки зрения описанный алгоритм оказывается вполне работоспособным – он генерирует качественные маршруты резки за приемлемое время, однако это эмпирический результат, полученный его применением к раскройным картам. Интересным является теоретическое обоснование свойств получаемых маршрутов. Наибольшую сложность представляет, конечно, третий шаг алгоритма – дискретная оптимизация, причём как с теоретический, так и с практической точки зрения. В данной статье рассматривается второй шаг алгоритма – непрерывная оптимизация. Для неё получается сформулировать некоторые утверждения о качестве получаемого решения.

Итак, далее мы предполагаем, что порядок обхода контуров задан и мы нашли положения всех точек врезки и каждое из них по отдельности доставляет локальный минимум: при бесконечно малом сдвиге любой одной из них, полная длина холостого хода увеличивается.

Кроме того, мы везде будем предполагать, что контуры состоят только из отрезков прямых (то есть являются многоугольниками).

## Локальный минимум

В этих условиях докажем, что найденное на шаге 2 решение доставляет локальный минимум, или в строгой формулировке: если сдвинуть несколько точек врезки , так чтобы они оставались на тех же звеньях контуров, то длина полученной ломаной не уменьшится.

Если мы сдвигаем несмежные вершины ломаной, то это утверждение тривиально следует из того, что каждая отдельная точка врезки находится в оптимальном положении. Её вклад в общую сумму

это функция с одним минимумом (если сама точка движется по прямой) и при любом сдвиге от минимума только увеличивается.

Рассмотрим поэтому простейший нетривиальный случай, когда мы сдвигаем две соседних точки врезки и . Передвинем их в новое положение и и докажем, что .

Предположим противное, путь имеются такие положения  и , что . Ясно, что одновременно и . Введём две переменных , задающих две новых точки и аналогично . Рассмотрим функцию , для неё по предположению выполнено условие

Так как мы считаем, что по отдельности точки и дают минимум полного расстояния, то . Если хотя бы одна из этих производных отлична от нуля, то рассмотрим функцию , для неё и значит d. Поэтому , такое что . Но по нашему предположению .

Осталось заметить, что функция выпукла вниз. Действительно, она представляет собой сумму трёх слагаемых вида (где – некоторые константы, зависящие от координат точек ), и для каждого из них вторая производная . Значит и .

Но выпуклая вниз функция не может принимать в середине интервала значение большее, чем на его концах и значит в этом случае наше предположение опровергнуто.

Если же , то, например из равенства следует, что либо , где за обозначено отражение точки относительно сегмента (на котором лежит ), см. рис. 4. Аналогично . Значит, обе точки и лежат на одном отрезке ([ или или или ), и если их сдвинуть с этого отрезка, расстояние только возрастёт.

Ci

Mi-1

Mi+1

Mi

Ci

M\*i-1

Mi+1

Mi

Mi-1

Ci

Mi-1

Mi+1

Mi

Mi+2

Ci+1

Рис 4. Случай

Таким образом, в обоих случаях наше предположение опровергнуто и значит, действительно, сдвигая точки врезки и в пределах содержащих их отрезков, мы можем только увеличить суммарную длину холостого хода.

Случай одновременного сдвига большего количества точек (находящихся подряд в маршруте обхода) рассматривается аналогично с увеличением размера выкладок.

## Достаточное условие глобального минимума

Попробуем распространить только что сделанное рассуждение на случай произвольных сдвигов точек и в пределах содержащих их контуров. Понятно, что в общем случае мы не можем гарантировать глобальную оптимальность такого маршрута. Например, на рис. 5 изображены два маршрута, каждый из них локально оптимален в использованном выше смысле, но зелёный короче красного и только он доставляет глобальный минимум.

C1

C2

M0

M3

Рис 5. Два локально оптимальных маршрута резки

Мы можем рассуждать так же, как в предыдущем шаге, только теперь вектор не направлен по сегменту контура . Однако, если случится так, что , то остальной ход доказательства можно повторить практически без изменений. Нам удалось сформулировать сравнительно простые условия, которые гарантируют неотрицательность производной (см. рис. 6):

* Отрезок пересекает контур ( и )
* Касательная в точке к эллипсу с фокусами и (проходящему через ), разделяет этот эллипс и контур ()

Mi-1

Mi+1

Mi

Ci

Ci

Mi

Mi+1

Mi-1

Рис 6. Достаточное условие глобальной минимальности маршрута

Эти условия легко проверяются программно для всех точек врезки построенного маршрута. Более того, при некотором навыке их можно быстро оценивать и простым взглядом на полученный маршрут. Более сложный второй вариант условия (с эллипсом) в значительной степени покрывается следующими соображениями (см. рис. 7):

* Если точка врезки – внутренняя точка звена контура , то это звено и является касательной к эллипсу (иначе бы мы нашли другое положение для на шаге 2). Достаточное условие оптимальности выполняется, если весь контур лежит по одну сторону прямой звена
* Если точка врезки – угловая точка контура (принадлежит двум звеньям), до достаточное условие будет выполнено, если весь контур находится внутри угла, образованного этими двумя звеньями
* И наконец, если область , ограниченная контуром выпуклая, то достаточное условие тоже соблюдается. В частности, если все контуры ограничивают выпуклые фигуры, то построенный маршрут всегда оптимален

Mi-1

Mi+1

Mi

Ci

Ci

Mi

Mi-1

Mi+1

Ci

Mi-1

Mi+1

Mi

Рис 7. Частные случаи достаточного условия

# Численные эксперименты

Оценка качества решений описанного алгоритма проводилась на нескольких раскройных планах, содержащих реальные детали. В качестве базы сравнения использовался алгоритм (см.) решения задачи GTSP, который даёт точное решение при количестве контуров .

На рис. 8 приведено точное решение, видны фиксированные положения точек врезки. На рис. 9 показано решение задачи CCP для того же раскройного плана.

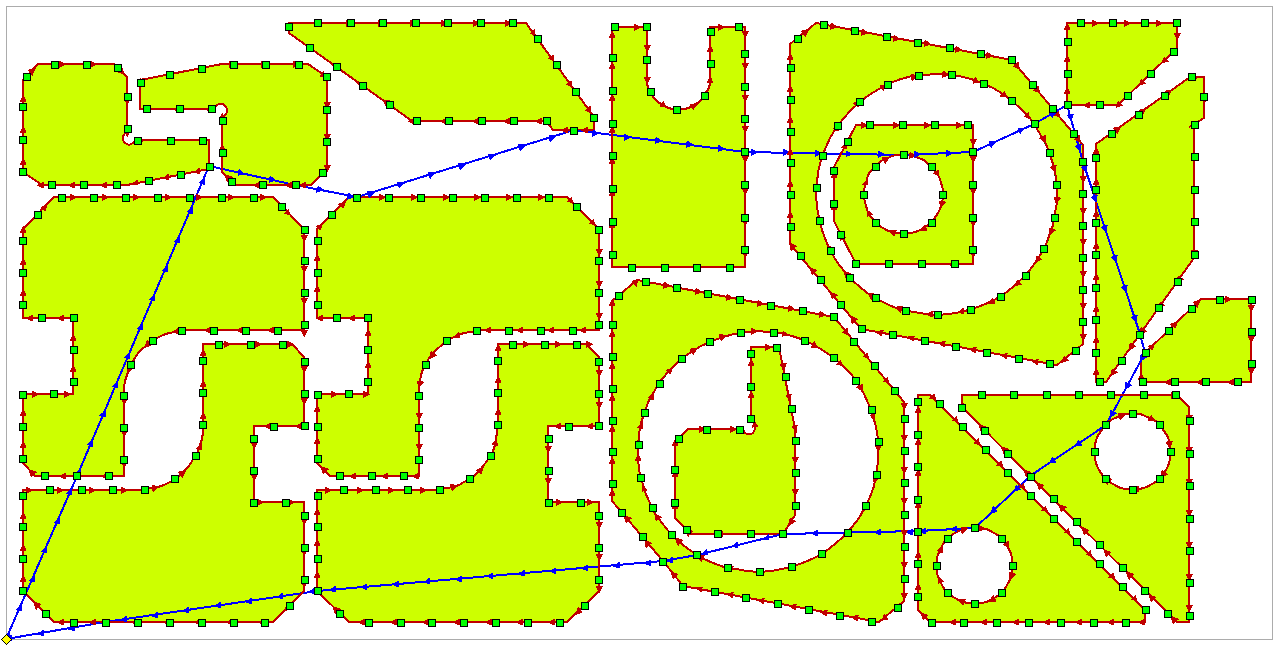


Рис 8. Точное решение задачи GTSP

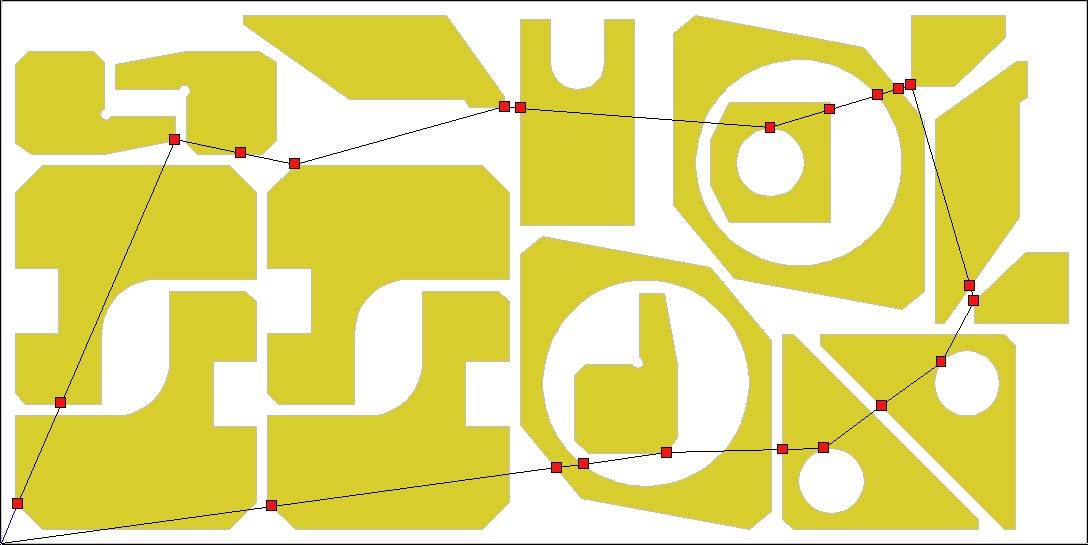


Рис 9. Решение задачи CCP

Видно, что оба алгоритма построили практически идентичный маршрут. Основное отличие вызвано процессом дискретизации для получения задачи GTSP. Из-за этого участки маршрута, прямые в решении CCP, оказываются слегка изломанными в решении GTSP, и суммарная длина холостого хода оказывается незначительно большей. Численно это видно в таблице 1.

| **Задание** | **№ 229** | **№ 464** | **№ 3211** |
| --- | --- | --- | --- |
| Кол-во деталей | 11 | 14 | 17 |
| Кол-во контуров | 12 | 21 | 22 |
| Периметр всех контуров, м | 24.609 | 21.717 | 25.051 |
| Кол-во точек в задаче GTSP | 491 | 429 | 493 |
| Холостой ход GTSP, м | 7.729 | 4.743 | 4.557 |
| Холостой ход CCP, м | 7.727 | 4.706 | 4.536 |

Таблица 1. Сравнение качества решения задач GTSP и CCP

# Заключение

1. Показано, что задача минимизации холостого инструмента машин листовой резки с ЧПУ для задачи маршрутизации из класса ССP может быть редуцирована до задачи без условий предшествования, что обеспечивает сокращение числа контуров и время работы алгоритма.
2. Предложен эвристический алгоритм решения задачи CCP, не использующий дискретизацию контуров.
3. Для любой фиксированной последовательности обхода контуров разработан эффективный алгоритм получения локального экстремума и описаны условия, при которых данный локальный экстремум является глобальным минимумом.

В дальнейшем предполагается реализация алгоритма для общего случая, когда точка врезки лежит вне контура в соответствии с технологическими требования листовой резки.

# Библиографический список