Московский государственный институт электроники и математики

(Технический университет)

Кафедра ИТАС

РЕФЕРАТ

Привести методы и алгоритмы решения задач компоновки, размещения и трассировки, возникающих в процессе конструирования. Оценить трудоемкость методов и качество получаемых с их помощью решений, рассматривая указанные выше задачи как задачи математического программирования.

 Выполнил:

 Проверил:

 Принял:

МОСКВА 2003

ОГЛАВЛЕНИЕ

1. Введение
2. Алгоритмы компоновка
3. Алгоритмы размещения
4. Алгоритмы трассировки
5. ВВЕДЕНИЕ

При конструкторском проектировании РЭА (радиоэлектронной аппаратуры) решаются задачи, связанные с поиском наилучшего варианта конструкции, удовлетворяющего требованиям технического задания и максимально учитывающего возможности технологической базы производства. Тесная взаимосвязанность задач и большая размерность каждой из них обычно не позволяет предложить метод поиска оптимального конструктивного решения в едином цикле в связи с трудностями создания общей математической модели, комплексно учитывающей особенности конструкторско-технологической базы производства. Поэтому разработка и реализация алгоритмов и методов решения отдельных задач этапа конструкторского проектирования: компоновки, размещения и трассировки,- до сих пор остаются актуальными проблемами, решение которых неотъемлемо связано с развитием систем автоматизации проектирования.

1. АЛГОРИТМЫ КОМПОНОВКИ

На этапе конструкторского проектирования решаются вопросы, связанные с компоновкой элементов логической схемы в модули, модулей в ячейки, ячеек в панели и т. д. Эти задачи в общем случае тесно связаны между собой, и их решение позволяет значительно сократить затраты и трудоемкость указанного этапа в САПР. Обычно задачи компоновки рассматриваются как процесс принятия решений в определенных или неопределенных условиях, в результате выполнения которого части логической схемы располагаются в конструктивных элементах i-го уровня, а эти элементы размещаются в конструктивных элементах (i+1) –го уровня и т. д., причем расположение выполняется с оптимизацией по выбранному критерию.

Компоновкой электрической схемы РЭА на конструктивно законченные части называется процесс распределения элементов низшего конструктивного уровня в высший в соответствии с выбранным критерием. Основным для компоновки является критерий электромагнитотепловой совместимости элементов низшего уровня. Данный критерий определяет область допустимых разбиений схемы, на которой формулируются другие критерии. Такими критериями могут быть: минимум типов конструктивно законченных частей, плотность компоновки, минимум соединений между устройствами, простота диагностирования и др. Очевидно, что внешние соединения между частями схем являются одним из важнейших факторов, определяющих надежность РЭА. Поэтому наиболее распространенным критерием является критерий минимума числа внешних связей. Выполнение этого критерия обеспечивает минимизацию взаимных наводок, упрощение конструкции, повышение надежности и т. д.

 Для построения формальной математической модели компоновочных задач удобно использовать теорию графов. При этом электрическую схему интерпретируют ненаправленным мультиграфом, в котором каждому конструктивному элементу (модулю) ставят в соответствие вершину мультиграфа, а электрическим связям схемы – его ребра. Тогда задача компоновки формулируется следующим образом, Задан мультиграф G(X,U). Требуется “разрезать” его на отдельные куски G1(X1,U1), G2(X2,U2),…, Gk(Xk,Uk) так, чтобы число ребер, соединяющих эти куски, было минимальным, т.е.

минимизировать *i≠j*

при i,j = 1,2,…,k,

где U*ij* – множество ребер, соединяющих куски G*i*(X*i*,U*i*) и G*j*(X*j*,U*j*).

 Другими словами разбиениями частей совокупности G на графы считаются, если любая часть из этой совокупности не пустая; для любых двух частей пересечение множества ребер может быть не пустым; объединение всех частей в точности равно графу G.

 Известные алгоритмы компоновки можно условно разбить на пять групп:

1. алгоритмы, использующие методы целочисленного программирования.
2. последовательные алгоритмы
3. итерационные алгоритмы
4. смешанные алгоритмы

Алгоритмы первой группы хотя и позволяют получить точное решение задачи, однако для устройства реальной сложности фактически не реализуемы на ЭВМ. В последнее время наибольшее распространение получили приближенные алгоритмы компоновки (последовательные, итерационные, смешанные). При использовании последовательных алгоритмов сначала по определенному правилу выбирают вершину графа, затем осуществляют последовательный выбор вершин (из числа нераспределенных) и присоединение их к формируемому куску графа. После образования первого куска переходят ко второму и т. д. до получения желаемого разрезания исходного графа. В итерационных алгоритмах начальное разрезание графа на куски выполняют произвольным образом; оптимизация компоновки достигается парными или групповыми перестановками вершин графа из различных кусков. Процесс перераспределения вершин заканчивают при получении локального экстремума целевой функции, удовлетворяющим требованиям разработчика. В смешанных алгоритмах компоновки для получения начального варианта “разрезания” используется алгоритм последовательного формирования кусков; дальнейшая оптимизация решения осуществляется перераспределением вершин между отдельными кусками графа.

Последовательные алгоритмыкомпоновки

В последовательных алгоритмах компоновки «разрезание» исходного графа G(X,U) на куски G1(X1,U1), G2(X2,U2),…, Gk(Xk,Uk) сводится к следующему. В графе G(X,U) находят вершину x*i* X с минимальной локальной степенью .

Если таких вершин несколько, то предпочтение отдают вершине с максимальным числом кратных ребер. Из множества вершин, смежных с вершинами формируемого куска графа G1(X1,U1), выбирают ту, которая обеспечивает минимальное приращение связей куска с еще нераспределенными вершинами. Данную вершину x*i* X \ X1 включают в G1(X1,U1), если не происходит нарушения ограничения по числу внешних связей куска, т.е.

,

где α*jε* – элемент матрицы смежности исходно графа G(X,U); δ(x*g*) – относительный вес вершины x*g,* , равный приращению числа внешних ребер куска G1(X1,U1) при включении вершины x*g*  во множество X1; E – множество индексов вершин, включенных в формируемый кусок графа на предыдущих шагах алгоритма; m – максимально допустимое число внешних связей отдельно взятого куска со всеми оставшимися.

 Указанный процесс продолжается до тех пор, пока множество X1 не будет содержать n элементов либо присоединение очередной нераспределенной вершины x*j*  к куску G1(X1,U1) не приведет к нарушению ограничения по числу внешних соединений куска, равному

 Следует отметить, что величина не является монотонной функцией |X1|, поэтому, для того чтобы убедится в невозможности дальнейшего формирования куска вследствие нарушения последнего ограничения, необходимо проверить его невыполнимость на последующих шагах увеличения множества X1 вплоть до n. В качестве окончательного варианта выбирают кусок G10(X10,U10), содержащий максимально возможное число вершин графа G(X,U), для которого выполняются ограничения на число внешних связей и входящих в него вершин (nmin-nmax).

 После преобразования куска G10(X10,U10) процесс повторяют для формирования второго, третьего и т.д. кусков исходного графа с той лишь разницей, что рассмотрению подлежат вершины, не вошедшие в предыдущие куски.

 Сформулируем алгоритм последовательной компоновки конструктивных элементов.

1. t:0
2. X*f* = X*t* = Ø; t=t+1; Θ=1; α=nmax,

Где t, Θ – порядковые номера формируемого куска и присоединяемой вершины; α – ограничение на число вершин в куске.

1. По матрице смежности исходного графа | α*hp*|NxN, где N – число вершин исходного графа (при большом значении N для сокращения объема оперативной памяти ЭВМ используем не саму матрицу смежности, а её кодовую реализацию), определяем локальные степени вершин .

1. Из множества нераспределенных вершин X выбираем вершину x*j* с ρ(x*j*) = . Переходим к п.6. Если таких вершин несколько, то переходим к п.5

1. Из подмножества вершин X*l* с одинаковой локальной степенью выбирают вершину x*j* с максимальным числом кратных ребер (минимальным числом смежных вершин), т.е. |Гx*j*| = .

1. Запоминаем исходную вершину формируемого куска графа . Переходим к п.10

1. По матрице смежности строим множество X*s* = и определяем относительные веса вершин :

.

1. Из множества X*S*  выбираем вершину . Переходим к п.10. Если таких вершин несколько, то переходим к п.9.

1. Из подмножества вершин X*v* с одинаковым относительным весом выбираем вершину x*j* с максимальной локальной степенью, т.е. .

1. .

1. Если >m , то переходим к п.13.
2. Рассмотренные вершины включаем в формируемый кусок X*f* = X*t* .
3. Θ = Θ + 1.
4. Если Θ> α, то переходим к п.15, а противном случае – к п.7.
5. Если |X*f*|<nmin, где nmin – минимально допустимое число вершин в куске, то переходим к п.21.
6. Выбираем окончательный вариант сформированного куска графа:

X*t* = X*f* ; X = X \ X*t* ; α = nmax .

1. Если |X|> nmax , то переходим к п.20.
2. Если |X|< nmin , то переходим к п.21.
3. Определяем число внешних связей последнего куска графа:

 ,

 где F – множество индексов вершин, входящих в X. Если , то переходим к п.21, в противном случае – к п.24.

1. Если t<k – 1 , где k - число кусков разрезания графа, то переходим к п.2, в противном случае – к п.23.
2. Предыдущий цикл «разрезания» считаем недействительным. Если t>1, т.е. имеется как минимум один ранее сформированный кусок, то переходим к п.22. в противном случае – к п.23.
3. Ищем другой допустимый вариант формирования предыдущего куска с меньшим числом вершин: t = t – 1; .

Переходим к п.7.

1. Задача при заданных ограничениях не имеет решения.
2. Конец работы алгоритма.

Рассмотренный алгоритм прост, легко реализуется на ЭВМ и позволяет получить решение задачи компоновки. Также среди достоинств данной группу алгоритмов выступает высокое быстродействие их при решении задач компоновки.

 Основным недостатком последовательного алгоритма является неспособность находить глобальный минимум количества внешних связей (не анализируются возможные ситуации). Наибольшая эффективность метода последовательного разбиения графа имеет место, когда число вершин графа G значительно больше вершин в любой части разбиения.

Итерационные алгоритмы компоновки

 Сущность данной группы алгоритмов заключается в выборе некоторого начального «разрезания» исходного графа на куски (вручную или с помощью последовательного метода компоновки) и последующего его улучшения с помощью итерационного парного или группового обмена вершин из различных кусков. При этом для каждой итерации осуществляется перестановка тех вершин, которая обеспечивает максимальное уменьшение числа связей между кусками графа или максимальное улучшение другого выбранного показателя качества с учетом используемых ограничений (например, на максимальное число внешних ребер любого отдельно взятого куска).

 Рассмотрим основную идею итерационного алгоритма разбиения графа G, заданного матрицей смежности, с минимизацией числа соединительных ребер. Разбиение графа G = (X,U) на *l* подграфов G1 = (X1,U1), G2 = (X2,U2),…,G*l* = (X*l*,U*l)* сведем к разбиению на два подграфа. С этой целью в матрице смежности R выделим по главной диагонали две подматрицы R1 и R2. При этом порядок подматрицы R1 равен числу вершин, которые должны находится в G1, а порядок подматрицы R2 – числу всех оставшихся вершин графа. Необходимо так переставить строки и столбцы матрицы R, чтобы число ребер между G1 и оставшейся частью графа G было минимальным. После этого подматрицу R1 из матрицы R исключаем, вычеркнув из R строки и столбцы, соответствующие элементам R1. Далее подматрицу R1’ разбиваем снова на две подматрицы R2 и R2’, причем порядок R2 соответствует числу вершин второго выделяемого подграфа, а порядок R2 – числу оставшихся вершин графа. Переставляем строки и столбцы R1’ с целью минимизации числа соединительных ребер. После этого подматрица R2’ исключается и процесс повторяется до тех пор, пока не будет выполнено разбиение графа G на *l* подграфов.

 Основная идея алгоритма заключается в выборе таких строк и столбцов, перестановка которых приводит к сосредоточению в диагональных клетках матрицы R максимального числа элементов. Построим прямоугольную матрицу W = ||*wi,j||n*ix(*n-ni*), в которой строки определяются вершинами из множества *I*, а столбцы – из множества *V*, . На пересечении *k* строки (и *q* столбца находится элемент

 ,

где *rk,q* – элемент матрицы смежности R.

 Элемент *wk,q* матрицы *W*  характеризует изменение числа соединительных ребер между G*i* и G*j* при перестановке вершин и . Используя матрицу *W* , можно найти подстановку, которая увеличит число элементов в подматрицах R1 и R1’ . Такой процесс повторяется до тех пор, пока в подматрице R1 не сосредоточится максимальное число единиц.

 В итерационных алгоритмах предусмотрена возможность поиска оптимального варианта для различных начальных разбиений. Это связано с тем, что при использовании итерационных алгоритмов оптимальность решения в значительной мере зависит от того, насколько удачно было произведено начальное разбиение графа G(X,U).

 Итерационные алгоритмы компоновки обеспечивают высокое качество решения задачи, однако требуют больших затрат машинного времени, чем последовательные алгоритмы. Для сокращения числа итераций обмена вершин между кусками в смешанных алгоритмах для получения начального «разрезания» графа применяют последовательные методы формирования его кусков. С этой целью в некоторых итерационных алгоритмах используют процесс групповой перестановки взаимно непересекающихся пар вершин.

1. АЛГОРИТМЫ РАЗМЕЩЕНИЯ

Исходной информацией при решении задач размещения являются: данные о конфигурации и размерах коммутационного пространства, определяемые требованиями установки и крепления данной сборочной единицы в аппаратуре; количество и геометрические размеры конструктивных элементов, подлежащих размещению; схема соединений, а также ряд ограничений на взаимное расположение отдельных элементов, учитывающих особенности разрабатываемой конструкции. Задача сводится к отысканию для каждого размещаемого элемента таких позиций, при которых оптимизируется выбранный показатель качества и обеспечивается наиболее благоприятные условия для последующего электрического монтажа. Особое значение эта задача приобретает при проектировании аппаратуры на печатных платах.

Основная сложность в постановке задач размещения заключается в выборе целевой функции. Связано это с тем, что одной из главных целей размещения является создание наилучших условий для дальнейшей трассировки соединений, что невозможно проверить без проведения самой трассировки. Любые другие способы оценки качества размещения (минимум числа пересечений ребер графа, интерпретирующего электрическую схему соединений, разбиение графа на минимальное число плоских суграфов и т.д.), хотя и позволяют создать благоприятные для трассировки условия, но не гарантируют получение оптимального результата, поскольку печатные проводники представляют собой криволинейные отрезки конечной ширины, конфигурация которых определяется в процессе их построения и зависит от порядка проведения соединений. Следовательно, если для оценки качества размещения элементов выбрать критерий, непосредственно связанный с получением оптимального рисунка металлизации печатной платы, то конечный результат может быть найден только при совместном решении задач размещения, выбора очередности проведения соединений и трассировки, что практически невозможно вследствие огромных затрат машинного времени.

Поэтому все применяемые в настоящее время алгоритмы размещения используют промежуточные критерии, которые лишь качественно способствуют решению основной задачи: получению оптимальной трассировки соединений. К таким критериям относятся: 1) минимум суммарной взвешенной длины соединений; 2) минимум числа соединений, длина которых больше заданной; 3) минимум числа пересечение проводников; 4) максимальное число соединений между элементами, находящимися в соседних позициях либо в позициях, указанных разработчиком; 5) максимум числа цепей простой конфигурации.

Наибольшее распространение в алгоритмах размещения получил первый критерий, что объясняется следующими причинами: уменьшение длин соединений улучшает электрические характеристики устройства, упрощает трассировку печатных плат; кроме того, он сравнительно прост в реализации.

В зависимости от конструкции коммутационной платы и способов выполнения соединений расстояние между позициями установки элементов подсчитывается по одной из следующих формул:

, ,

В общем виде задача размещения конструктивных элементов на коммутационной плате формулируется следующим образом. Задано множество конструктивных элементов R={*r1,r2,…,rn*} и множество связей между этими элементами *V=*{*v1,v2,…,vp*}, а также множество установочных мест (позиций) на коммутационной плате *T=*{*t1,t2,…,tk*}. Найти такое отображение множества R на множестве *T*, которое обеспечивает экстремум целевой функции

, где *cij –* коэффициент взвешенной связности.

Силовые алгоритмы размещения

В основу этой группы алгоритмов положен динамический метод В.С. Линского. Процесс размещения элементов на плате представляется как движение к состоянию равновесия системы материальных точек (элементов), на каждую из которых действуют силы притяжения и отталкивания, интерпретирующие связи между размещаемыми элементами. Если силы притяжения, действующие между любыми двумя материальными точками *ri* и *rj* пропорциональны числу электрических связей между данными конструктивными элементами, то состояние равновесия такой системы соответствует минимуму суммарной длины всех соединений. Введение сил отталкивания материальных точек друг от друга и от границ платы исключает возможность слияния двух любых точек и способствует их равномерному распределению по поверхности монтажного поля. Чтобы устранить возникновение в системе незатухающих колебаний, вводят силы сопротивления среды, пропорциональные скорости движения материальных точек.

 Таким образом, задача оптимального размещения элементов сводится к нахождению такого местоположения точек, при котором равнодействующие всех сил обращаются в нуль.

 К достоинствам данного метода относятся возможность получения глобального экстремума целевой функции, а также сведение поиска к вычислительным процедурам, для которых имеются разработанные численные методы.

 Недостатками являются трудоемкость метода и сложность его реализации (подбора коэффициентов для силовых связей); необходимость фиксирования местоположения некоторого числа конструктивных элементов на плате для предотвращения большой неравномерности их размещения на отдельных участках платы.

Итерационные алгоритмы размещения

Итерационные алгоритмы имеют структуру, аналогичную итерационным алгоритмам компоновки, рассмотренным ранее. В них для улучшения исходного размещения элементов на плате вводят итерационный процесс перестановки местами пар элементов.

 В случае минимизации суммарной взвешенной длины соединений формула для расчета изменения значения целевой функции при перестановке местами элементов *ri* и *rj* , закрепленных в позициях *tf* и *tg*, имеет вид:

 ,

где *p* и *h(p)* – порядковый номер и позиция закрепления неподвижного элемента *rp*. Если , то осуществляют перестановку *ri* и *rj* , приводящую к уменьшению целевой функции на , после чего производят поиск и перестановку следующей пары элементов и т.д. Процесс заканчивается получением такого варианта размещения, для которого дальнейшее улучшение за счет парных перестановок элементов невозможно.

 Использование описанного направленного перебора сокращает число анализируемых вариантов размещения (по сравнению с полным перебором), но приводит к потере гарантии нахождения глобального экстремума целевой функции.20

 Алгоритмы дано группы характеризуются достаточно высоким быстродействием. Алгоритмы с групповыми перестановками элементов на практике используются редко ввиду их сложности, которая часто не оправдывает достигаемую степень улучшения результата.

Последовательные алгоритмы размещения

Последовательные алгоритмы основаны на допущении, что для получения оптимального размещения необходимо в соседних позициях располагать элементы, максимально связанные друг с другом. Сущность этих алгоритмов состоит в последовательном закреплении заданного набора конструктивных элементов на коммутационной плате относительно ранее установленных. В качестве первоначально закрепленных на плате элементов обычно выбирают разъемы, которые искусственно «раздвигают» до краев платы. При этом все контакты разъемов равномерно распределяются по секциям (столбцам и строкам координатной сетки). На каждом *l*-ом шаге (*l*=1,2,…,*n*) для установки на коммутационную плату выбирают элемент из числа еще не размещенных, имеющий максимальную степень связности с ранее закрепленными элементами *Rl-1*. В большинстве используемых в настоящее время алгоритмов оценку степени связности производят по одной из следующих формул:

;

,

где *cij* – коэффициент взвешенной связности элементов *i* и *j*; *Jl-1* – множество индексов элементов, закрепленных на предыдущих *l-1* шагах; *n* – общее число размещенных элементов.

 Если установочные размеры всех размещаемых на плате элементов одинаковы, то выбранный на очередном шаге элемент закрепляют в той позиции из числа незанятых, для которой значение целевой функции с учетом ранее размещенных элементов *Rl-1* минимально. В частности, если критерием оптимальности является минимум суммарной взвешенной длины соединений, то

,

где *dfj* – расстояние между *f*-ой позицией установки элемента и позицией размещенного ранее элемента *rj*; *Tl-1* – множество позиций, занятых элементами после (*l-1*)-го шага алгоритма.

 Процесс размещения алгоритма заканчивается после выполнения *n* шагов алгоритма.

 Алгоритмы, использующие последовательный процесс закрепления элементов в позициях, являются в настоящее время самыми быстродействующими. Однако по качеству получаемого решения последовательные алгоритмы уступают итерационным. Поэтому их используют обычно для получения начального размещения элементов на плате.

1. АЛГОРИТМЫ ТРАССИРОВКИ

 Трассировка соединений является, как правило, заключительным этапом конструкторского проектирования РЭА и состоит в определении линий, соединяющих эквипотенциальные контакты элементов, и компонентов, составляющих проектируемое устройство.

 Задача трассировки – одна из наиболее трудоемких в общей проблеме автоматизации проектирования РЭА. Это связано с несколькими факторами, в частности с многообразием способов конструктивно-технологической реализации соединений, для каждого из которых при алгоритмическом решении задачи применяются специфические критерии оптимизации и ограничения. С математической точки зрения трассировка – наисложнейшая задача выбора из огромного числа вариантов оптимального решения.

 Одновременная оптимизации всех соединений при трассировке за счет перебора всех вариантов в настоящее время невозможна. Поэтому разрабатываются в основном локально оптимальные методы трассировки, когда трасса оптимальна лишь на данном шаге при наличии ранее проведенных соединений.

 Основная задача трассировки формулируется следующим образом: по заданной схеме соединений проложить необходимые проводники на плоскости (плате, кристалле и т.д.), чтобы реализовать заданные технические соединения с учетом заранее заданных ограничений. Основными являются ограничения на ширину проводников и минимальные расстояния между ними.

 Исходной информацией для решения задачи трассировки соединений обычно являются список цепей, параметры конструкции элементов и коммутационного поля, а также данные по размещению элементов. Критериями трассировки могут быть процент реализованных соединений, суммарная длина проводников, число пересечений проводников, число монтажных слоев, число межслойных переходов, равномерность распределения проводников, минимальная область трассировки и т.д. Часто эти критерии являются взаимоисключающими, поэтому оценка качества трассировки ведется по доминирующему критерию при выполнении ограничений по другим критериям либо применяют аддитивную или мультипликативную форму оценочной функции, например следующего вида

, где *F* – аддитивный критерий; *λi* – весовой коэффициент; *fi* – частный критерий; *p* – число частных критериев.

 Известные алгоритмы трассировки печатных плат можно условно разбить на три большие группы:

1. Волновые алгоритмы, основанные на идеях Ли и разработанные Ю.Л. Зиманом и Г.Г. Рябовым. Данные алгоритмы получили широкое распространение в существующих САПР, поскольку они позволяют легко учитывать технологическую специфику печатного монтажа со своей совокупностью конструктивных ограничений, Эти алгоритмы всегда гарантируют построение трассы, если путь для нее существует;
2. Ортогональные алгоритмы, обладающие большим быстродействием, чем алгоритмы первой группы. Реализация их на ЭВМ требует в 75-100 раз меньше вычислений по сравнению с волновыми алгоритмами. Такие алгоритмы применяют при проектировании печатных плат со сквозными металлизированными отверстиями. Недостатки этой группы алгоритмов связаны с получением большого числа переходов со слоя на слой, отсутствием 100%-ой гарантии проведения трасс, большим числом параллельно идущих проводников;
3. Алгоритмы эвристического типа. Эти алгоритмы частично основаны на эвристическом приеме поиска пути в лабиринте. При этом каждое соединение проводится по кратчайшему пути, обходя встречающиеся на пути препятствия.

Волновой алгоритм Ли

 Данный алгоритм является классическим примером использования методов динамического программирования для решения задач трассировки печатных соединений. Основные принципы построения трасс с помощью динамического алгоритма сводятся к следующему.

 Все ячейки монтажного поля подразделяют на занятые и свободные. Занятыми считаются ячейки, в которых уже расположены проводники, построенные на предыдущих шагах, или находятся монтажные выводы элементов, а также ячейки, соответствующие границе платы и запрещенным для прокладывания проводников участкам. Каждый раз при проведении новой трассы можно использовать лишь свободные ячейки, число которых по мере проведения трасс сокращается.

 На множестве свободных ячеек коммутационного поля моделируют волну влияния из одной ячейки в другую, соединяемых впоследствии общим проводником. Первую ячейку, зарождается волна влияний, называют источником, а вторую – преемником волны. Чтобы иметь возможность следить за прохождением фронта волны влияний, его фрагментам на каждом этапе присваивают некоторые веса:

,

где *Pk* и *Pk-1* - веса ячеек *k*-го и (*k-1*)-го фронтов; - весовая функция, являющаяся показателем качества проведения пути, каждый параметр которой характеризует путь с точки зрения одного из критериев качества (длины пути, числа пересечений и т.п.). На *Pk* накладывают одно ограничение – веса ячеек предыдущих фронтов не должны быть больше весов ячеек последующих фронтов. Фронт распространяется только на соседние ячейки, которые имеют с ячейками предыдущего фронта либо общую сторону, либо хотя бы одну общую точку. Процесс распространения волны продолжается до тех пор, пока её расширяющийся фронт не достигнет приемника или на Θ-ом шаге не найдется ни одной свободной ячейки, которая могла бы быть включена в очередной фронт, что соответствует случаю невозможности проведения трассы при заданных ограничениях.

 Если в результате распространения волна достигла приемника, то осуществляют «проведение пути», которое заключается в движении от приемника к источнику про пройденным на этапе распространения волны ячейкам, следя за тем, чтобы значения *Pk* монотонно убывали. В результате получают путь, соединяющий эти две точки. Из описания алгоритма следует, что все условия, необходимые для проведения пути, закладываются в правила приписания веса ячейкам.

 Чтобы исключить неопределенность при проведении пути для случая, когда несколько ячеек имеют одинаковый минимальный вес, вводят понятие ***путевых координат***, задающих предпочтительность проведения трассы. Каждое направление кодируют двоичным числом по mod *q*, где *q* – число просматриваемых соседних ячеек. При этом чем более предпочтительно то или иное направление, тем меньший числовой код оно имеет. Например, если задаться приоритетным порядком проведения пути сверху, справа, снизу и слева, то коды соответствующих путевых координат будут 00, 01, 10, и 11. Приписание путевых координат производят на этапе распространения волны. При проведении пути движение от ячейки к ячейке осуществляют по путевым координатам.

 Существенными недостатками волнового алгоритма являются малое быстродействие и большой объем оперативной памяти ЭВМ, необходимый для хранения информации о текущем состоянии всех ячеек коммутационного поля, возможность построения лишь соединений типа «ввод-вывод». Попытки устранить указанные недостатки привели к созданию ряда модификаций волнового алгоритма.

Модификации алгоритма Ли

Метод встречной волны

В данном методе источниками волн являются обе ячейки, подлежащие электрическому объединению. При этом на каждом *k*-ом шаге поочередно строят соответствующие фронты первой и второй волн, распространяющихся из этих ячеек. Процесс продолжается до тех пор, пока какая-либо ячейка из фронта первой волны не попадет на фронт второй волны или наоборот. Проведение пути осуществляют из данной ячейки в направлении обоих источников по правилам, описанным в волновом алгоритме Ли.

 Оценим число ячеек, просматриваемых на этапе распространения волны, при использовании в качестве источников одной или двух объединяемых точек. Пусть расстояние между этими точками *R*. Тогда для первого случая в момент достижения волной ячейки-приемника площадь просмотренной окрестности имеет величину (знак равенства соответствует отсутствию преград пути распространения волны). Для второго случая в момент встречи фронтов двух волн площадь просмотренной окрестности .

 Таким образом, при использовании метода встречной волны просматриваемая площадь, а следовательно, и время, затрачиваемое на этапе распространения волны, уменьшаются примерно вдвое.

 Недостатком метода является необходимость выделения дополнительного разряда памяти на каждую рабочую ячейку поля для хранения информации о принадлежности её к первой или второй волне. Однако выигрыш в повышении быстродействия выполняет указанный недостаток, поэтому данный метод используют во всех случаях, когда это позволяет объем оперативной памяти ЭВМ.

Лучевой алгоритм трассировки

 В данном алгоритме, предложенным Л.Б. Абрайтисом, выбор ячеек для определения пути между соединяемыми точками A и B производят по заранее заданным направлениям, подобным лучам. Это позволяет сократить число просматриваемых алгоритмом ячеек, а следовательно, и время на анализ и кодировку их состояния, однако приводит к снижению вероятности нахождения пути сложной конфигурации, и усложняет учет конструктивных требований к технологии печатной платы.

 Работа алгоритма заключается в следующем. Задается число лучей, распространяемых из точек A и B, а также порядок присвоения путевых координат (обычно число лучей для каждой ячейки-источника принимается одинаковым). Лучи A(1), A(2),…, A(n) и B(1), B(2),…, B(n) считают одноименными, если они распространяются из одноименных источников A или B. Лучи A(i) и B(i) являются разноименными по отношению друг к другу. Распространение лучей производят одновременно из обоих источников до встречи двух разноименных лучей в некоторой ячейке C. Путь проводится из ячейки C и проходит через ячейки, по которым распространялись лучи.

 При распространении луча может возникнуть ситуация, когда все соседние ячейки будут заняты. В этом случае считается заблокированным и его распространение прекращается.

Лучи:

A(1): вверх, влево

A(2): влево, вверх

B(1): вниз, вправо

B(2): вправо, вниз

На втором шаге луч B(1) оказывается заблокированным, а на четвертом шаге блокируется и луч A(2). Лучи A(1) и B(2) встречаются в ячейке C на восьмом шаге.

 Обычно с помощью лучевого алгоритма удается построить до 70-80% трасс, остальные проводят, используя волновой алгоритм или вручную. Его применение особенно выгодно при проектировании плат с невысокой плотностью монтажа.

ИСПОЛЬЗУЕМАЯ ЛИТЕРАТУРА

1. Б.Н. Деньдобренко, А.С. Малика «Автоматизация конструирования РЭА»,

Москва «Высшая школа» 1980.

1. В.М. Курейчик «Математическое обеспечение конструкторского и технологического проектирования с применением САПР»,

Москва «Радио и связь» 1990.

1. К.К. Морзов, В.Г. Одиноков, В.М. Курейчик «Автоматизированное проектирование конструкций радиоэлектронной аппаратуры», Москва «Радио и связь» 1983.
2. В.Н. Ильин, В.Т. Фролкин, А.И. Бутко и др.; «Автоматизация схемотехнического проектирования: Учебное пособие для вузов», Москва «Радио и связь» 1987.