Wiring Knock-Knee Layouts A Global Approach[◇]

Majid Sarrafzadeh[†]
Frank Wagner[‡]
Dorothea Wagner[♯]
Karsten Weihe[♯]

B 92–07
March 1991

Abstract

We present a global approach to solve the three-layer wirability problem for knock-knee layouts. In general, the problem is $\mathcal{NP}$-complete. Only for very special layouts a polynomial three-layer wiring algorithm is known up to now. In this paper we show that for a large class of layouts the problem can be formulated as a path problem in a special class of graphs or as a two-satisfiability problem and thus may be solved efficiently. Moreover, it is shown that a minimum stretching of the layout into a layout belonging to this class can be found by solving a clique cover problem in an interval graph. This problem is polynomially solvable as well. Altogether, the method also yields a good heuristic to derive three-layer wirability for arbitrary knock-knee layouts.

[◇]Part of this work was done while Majid Sarrafzadeh, Dorothea Wagner and Frank Wagner were with the LEONARDO FIBONACCI INSTITUTE for the Foundations of Computer Science, Trento, Italy. Majid Sarrafzadeh also acknowledges the National Science Foundation for supporting this research in part under grant MIP-8921540. Dorothea Wagner and Karsten Weihe acknowledge the Deutsche Forschungsgemeinschaft for supporting this research under grant Mö 446/1-3.
[†]Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60208, USA.
[‡]Institut für Informatik, Fachbereich Mathematik, Freie Universität Berlin, Arnimallee 2-6, W-1000 Berlin 33, Germany.
[♯]Fachbereich Mathematik, Technische Universität Berlin, Straße des 17. Juni 136, W-1000 Berlin 12, Germany.

1 Introduction

Routing is an important problem encountered in the design of integrated circuits. After placement and global routing, in the detailed routing phase the course of the wires connecting the cells is determined. Since layouts containing *crossings* or *knock-knees* (points where two wires bend) cannot be realized in a single plane, normally the routing is carried out in two steps, the *layout* and the layer assigment called *wiring*. In case of knock-knee layouts the wiring, i.e., the conversion of a layout in the plane to an actual three dimensional configuration of wires to avoid contact between different wires, is a non-trivial task.

There have been made several contributions to this problem [1], [2], [8], [9], [11], [16], in general all based on a systematic approach developed by Lipski & Preparata in [15] and [12] for grid based layouts. This combinatorial framework is derived from the observation, that any wiring induces a partition of the layout into a two-colorable map containing diagonal partition lines induced by the knock-knees and additional vertical and horizontal partition lines. For wirability in two, three, four or more layers, equivalent conditions for the corresponding partitions are given. One consequence of this approach is that it is $\mathcal{NP}$-complete to decide if a given layout is wirable in three layers [11]. But every layout is easily wirable in four layers [1], [16]. Only very restricted layouts are two-layer wirable, hence in general wirability in three layers is the best one can expect for a given layout.

Essentially, there are two different approaches to attack this problem. One possibility is to go one step back within the design process and consider the routing problem itself, i.e. to aim for layouts that are provably three-layer wirable. For channel routing problems, algorithms that guarantee three-layer wirable layouts are given in [7], [10], [14], [15] and [18]. The proof of three-layer wirability strongly depends on the very special structure of these layouts, i.e. they contain at most two knock-knees per vertical line, and in case of two knock-knees these even lie in opposite directions.

The corresponding three-layer partitions contain only vertical partition edges in addition to the diagonals induced by knock-knees.

Such special layouts cannot be expected for more general routing problems resp. in most cases do not even exist.

The second approach is to transform a layout into a three-layer wirable layout by appropriate *stretchings* [2], [8], [9]. Stretching a layout increases the area required for the layout. Obviously, the problem to find a minimum stretching for three-layer wirability is $\mathcal{NP}$-hard. It remains $\mathcal{NP}$-hard for minimum stretching within only one dimension.

In this paper we consider the problem if for a given layout there is a three-layer partition containing only vertical or only horizontal partition edges (in addition to the diagonal edges induced by the knock-knees). This problem can be formulated as a path problem in a special graph or as a two-satisfiability problem, and may be decided in time at most linear in the layout area. Local layout modifications, such as those applied in [15] and [10], and the local introduction of the orthogonal dimension (e.g. local use of horizontal edges within the vertical approach) are contained as well. Moreover, the method yields the minimum number of vertical lines (resp. tracks) to be added to transform a layout into a stretched layout that admits a legal partition for three-layer wirability with only vertical (resp. horizontal) additional partition edges. In the most general case, this problem is equivalent to the minimum clique cover problem in interval graphs, which is solvable in time linear in the size of the graph [6].

2 Preliminaries

In this section we review the basic definitions and results from [12] concerning the wiring of knock-knee layouts.

Consider a layout in a rectilinear grid graph. A *conducting layer*, or simply *layer* is a graph isomorphic to the layout grid. Conducting layers $L_1, \ldots, L_k$ are assumed to be stacked on top of each other, with L_1 on the bottom and L_k on the top. A contact between two layers, called a *via*, can be placed only at a grid vertex.

A correct *layer assignment* or *wiring* of a given layout is a mapping of each edge of a wire to a layer, such that:

1. No two different wires share a vertex on the same layer.
2. If adjacent edges of a wire are assigned to different layers, a via is established between these layers at their common vertex.
3. If a via connects L_h and L_j $(h < j)$, then layers $L_i, h < i < j$, are not used at that vertex by any other wire.

To determine a correct wiring of a knock-knee layout only those grid vertices where two different wires share a vertex, i.e. cross or form a knock-knee, are of relevance. Denote the part of a layout induced by these grid vertices as the *core* of the layout. Then the following lemma holds.

Lemma 1 *[12] A layout is wirable in k layers iff its core is.*

The basic idea now is, that any correct wiring of a layout in a fixed number of layers induces a partition of the layout area into the following two types of regions.

- *V-region*: The region where vertical wire edges lie above horizontal ones.
- *H-region*: The region where horizontal wire edges lie above vertical ones.

By this partition, the layout can be considered as a *two-colorable map.*

Obviously, the entire unit square around a crossing in the layout belongs to one color region. Since two wires that form a knock-knee cannot change their relative position through their common grid vertex in a correct wiring, the unit square around a knock-knee must belong to both regions, say the "triangle" above the imaginary diagonal through the knock-knee to the V-region, and the triangle below to the H-region or vice-versa. Consequently, the set of partition edges P contains all diagonals through knock-knees and, in addition, "appropriate" vertical and horizontal edges of the dual grid. The following properties of two-colorable maps state necessary conditions how a partition inducing a correct wiring has to look like.

Lemma 2 *[12] A set P of diagonals and dual grid edges defines a two-colorable map iff*

1. *each interior vertex (of the dual of the routing graph) is incident with an even number of edges of P,*
2. *each connected component of the boundary of the layout core is incident with an even number of edges of P.*

The main result concerning the three-layer wirability problem can then be stated as follows.

Theorem 3 *[12] A layout is three-layer wirable iff there exists a partition into a two-colorable map containing none of the eight patterns shown in figure 1.*

By a *vertical elementary stretching* along a column $\vec{i}$ (between vertical line i and $i+1$) we denote the operation that modifies the layout by cutting it along $\vec{i}$, moving its pieces horizontally one unit apart and inserting a new horizontal wire edge where a wire was crossing $\vec{i}$. This means that all wires crossing $\vec{i}$ are "stretched" as well. A *horizontal elementary stretching* is defined analogously. Any sequence of horizontal and vertical elementary stretching operations is simply called a *stretching.*

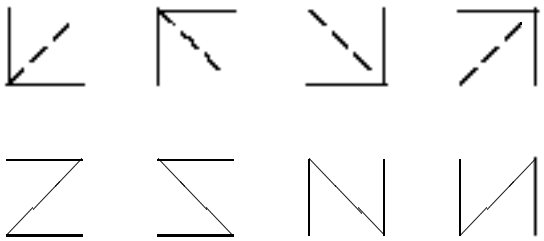

Figure 1: Forbidden patterns for a three-layer partition. Broken lines denote the absence of edges.

3 The New Wiring Approach

3.1 Constructing a legal partition

Consider the following problem.

- **Given** A layout.
- **Problem** Construct a legal three-layer partition containing only vertical (horizontal) partition edges additional to the diagonals through knock-knees. We call such a partition a *V-partition* (resp. *H-partition*.)

As stated in the last section, a layout admits a legal three-layer partition iff its core does. But we cannot conclude from this fact that a layout admits a V-partition (resp. H-partition) if its core does. (In fact, this is not true). So, we must really distinguish between the layout and its core.

We call a connected core component *V-convex* if the set of vertical edges of each vertical grid line covered by the core component forms one connected interval. (H-convex may be defined analogously.) For simplicity, let us first consider only layouts whose core is of V-convex shape.

A canonical approach to construct a V-partition is to scan the layout from left to right and add vertical partition edges in the dual grid depending on the knock-knees of the layout such that a two-colorable map without forbidden patterns arises. (An analogous method can be applied horizontally.)

The following lemma yields an efficient procedure to add vertical partition edges correctly.

Lemma 4 *Let P be a partition of a layout into a two-colorable map containing only vertical additional partition edges. Then for any vertical edge e of the grid dual to the layout grid, the state of e, i.e. $e \in P$ or $e \notin P$, uniquely determines the state of all vertical edges e' lying on the same vertical grid line as e.*

Proof

Consider a vertical line L (as a set of vertical edges) in the grid dual to the layout, and the diagonals induced by knock-knees that are incident to an edge of L. Call a vertex incident with one or three diagonals *odd*, all other vertices *even*. Then the set of odd vertices belonging to an edge of L partition L into intervals $I_1, \ldots, I_k$ say, of edges between two odd vertices resp. an odd vertex and the upper or lower boundary. For a vertical edge e of L, $e \in P$ induces that the interval, say I_j, containing e is completely contained in P, as well as every second interval above and below I_j. Precisely, for a vertical edge e' on L, $e' \in P$ iff e' belongs to an interval $I_{j+2 \cdot i}$ with $\lfloor \frac{-j}{2} \rfloor \leq i \leq \lfloor \frac{k-j}{2} \rfloor$.

Thus, for any vertical line of the grid dual to the layout, there are only two possibilities which vertical edges belong to a V-partition. The decision which possibilities can be chosen for one vertical line only depends on the state of its two neighbouring lines. (Look at the forbidden patterns.)

The problem of constructing a V-partition can be formulated as a *path problem* in a directed graph $G_p = (V, E_p)$ as follows. For each vertical line i in the grid dual to the layout, we introduce

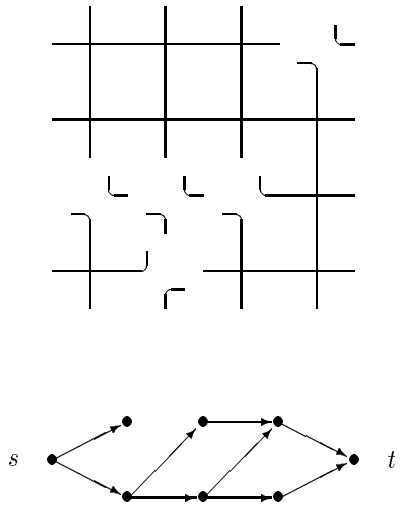

Figure 2: A layout core and its path graph.

two vertices v_i^0, v_i^1 corresponding to the two possible states (i.e. v_i^1 corresponding to "the uppermost edge belongs to P"). There are edges only between vertices corresponding to neighbouring vertical lines, where v_i^j, $v_{i+1}^k \in E_p$ iff there is no forbidden pattern between line i and line $i+1$ in case i has state j and $i+1$ has state k, for $j,k \in \{0,1\}$. In addition, we introduce a source s and a target t, and edges from s to the two vertices corresponding to the leftmost vertical line resp. from the two vertices corresponding to the rightmost vertical line to t.

We call G_p the *path graph* of the layout. Obviously, the time complexity for constructing the path graph is linear in the number of knock-knees in case the position of the knock-knees is given. See figure 2.

Corollary 5 *The problem to find a V-partition for a V-convex layout core is equivalent to finding an s-t-path in the corresponding path graph.*

Obviously, the time complexity for solving this path problem is linear in the spread of the layout. In case the knock-knee positions are given in order, the time complexity is even only linear in the number of knock-knees.

Now, consider the general case that the layout core consists of more than one component of arbitrary shape, possibly containing holes. First, we informally describe the situation. Observe that the components of the layout core may be considered independently of each other. For a vertical line covered by one component C, the edges contained in C are partitioned into maximal vertical intervals. In principle, there are two possible choices of vertical partition edges for each of these maximal intervals. Moreover, forbidden patterns can only appear between neighbouring intervals that overlap a common track. By that, the path graph for C arises canonically as a directed graph containing a set of pairs of vertices for each vertical line covered by C and possible edges between vertices corresponding to neighbouring intervals that overlap a common track. The problem to find a V-partition for C transforms to several path problems in the path graph that have to be solved simultaneously.

We think that it is more convenient to formulate the simultaneous solvability of these path problems as a *satisfiability problem*, where each clause consists of two literals (2SAT).

Let us first formulate the 2SAT problem corresponding to the simple path problem induced by a layout core of V-convex shape. We introduce a variable x_i for each pair of vertices v_i^0, v_i^1

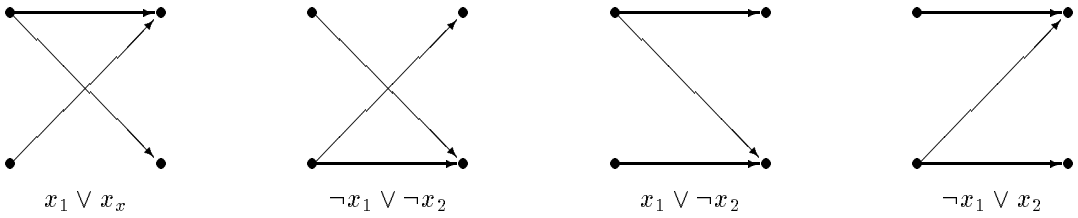

Figure 3: The clauses corresponding to state combinations.

(resp. vertical line i). For the set of possible edges from a v_i-vertex to a v_{i+1}-vertex (resp. state combinations), we have a set of at most four $\vee$-clauses containing x_i and x_{i+1}, negated or non-negated. Actually, we get one clause for each missing edge (resp. forbidden state combination). See figure 3. Then each s-t-path corresponds to a satisfying truth assignment for the set of all corresponding clauses.

In the general case of a core component C of non-convex shape, for example with holes, we have one variable for each maximal interval of a vertical line that belongs to C. The clauses correspond to all forbidden state combinations between neighbouring intervals overlapping a common track. Then the existence of a V-partition of C is equivalent to a truth assignment satisfying all corresponding clauses.

2SAT can be solved in time linear in the number of clauses [3]. Since the number of clauses for a layout is linear in the layout size, we obtain the following result.

Theorem 6 *For a layout, the existance of a V-partition can be decided in time linear in the layout size. If yes, a V-partition can be determined in linear time as well.*

3.2 Layout Modifications

There are even layouts containing at most two knock-knees per column, where the knock-knees are of opposite direction, as those guaranteed in [15] resp. [10], that do not admit a V-partition. [1]

In such a case however, in [15] resp. [10] V-partitions are constructed for slightly modified layouts. Such layout modifications can also be included into our approach.

A *local layout modification* can be applied to a layout, whenever two nets touch twice in the same column, or in two neighbouring columns. Then the layout can be transformed into an equivalent layout by replacing knock-knees by crossings and crossings by knock-knees. See figure 4.

Such a layout modification in one resp. two neighbouring columns induces a new partition of each of the related vertical lines into intervals between odd vertices. Thus, a possible layout modification can be introduced in the path graph as follows. For vertical lines $i, i+1$ (resp. $i, i+1, i+2$) involved in the modification join additional vertices $u_i^0, u_i^1, u_{i+1}^0, u_{i+1}^1, (u_{i+2}^0, u_{i+2}^1)$ and appropriate edges from v_{i-1}-vertices to u_i-vertices, from u_i-vertices to u_{i+1}-vertices and from u_{i+1}-vertices to v_{i+2}-vertices (resp. from u_{i+1}-vertices to u_{i+2}-vertices and from u_{i+2}-vertices to v_{i+3}-vertices). See figure 5.

In a similar way, the local use of horizontal partition edges may be involved.

3.3 Stretching for wirability

The problem of finding a minimum stretching of a layout such that the stretched layout is three-layer wirable is $\mathcal{NP}$-hard, even stretching within only one dimension. In this section we give a polynomial algorithm for the following problem.

- **Given** A layout.

[1] These examples have been found by applying the implemented method to layouts generated by the algorithms from [15] resp. [10]. Indeed, these examples are fairly large (spread 200). Because of lack of space, they are omitted.

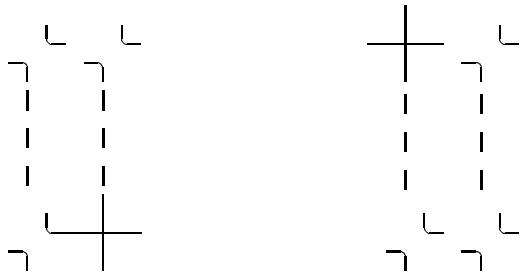

Figure 4: A layout modification.

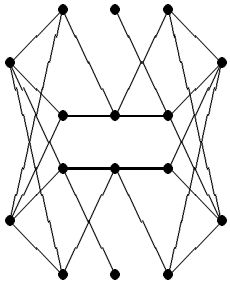

Figure 5: Part of the path graph corresponding to the layout modification in figure 4. The upper part corresponds to layout a) and the lower part to the modified layout shown in b).

- **Problem** Find a minimum stretching such that the stretched layout admits a V-partition (H-partition).

First, we restrict ourselves to the case that the core of the layout consists of only one component of V-convex shape. Then the method given in section 3.1 can be applied to this problem too. Consider the modification of the path graph $G_p = (V, E_p)$ induced by an elementary vertical stretching of the layout along column $\vec{i}$, i.e. by introducing a new vertical line, say r, between i and $i+1$. Then for the modified path graph $G'_p = (V', E'_p)$, two new vertices corresponding to r are contained, i.e. $V' = V \cup \{v_r^0, v_r^1\}$. All possible edges from v_i-vertices to v_r-vertices and from v_r-vertices to v_{i+1}-vertices are added, and all edges between v_i-vertices and v_{i+1}-vertices are deleted, i.e.

$$E'_p = E_p \backslash \{(v_i^j, v_{i+1}^k) : j,k \in \{0,1\}\} \cup \{(v_i^j, v_r^k), (v_r^k, v_{i+1}^l) : j,k,l \in \{0,1\}\}.$$

Thus, any directed path in G terminating with v_i^0 or v_i^1 is in G' connected to any directed path in G leaving v_{i+1}^0 or v_{i+1}^1. Generally, a stretching corresponds to transforming the path graph G to a new graph G', where directed paths in G are combined to an s-t-path in G' containing at least one v_i-vertex for all vertical lines i of the layout. It follows that a minimum stretching to guarantee a V-partition is equivalent to a minimum number of directed paths in G that can be combined to an s-t-path containing v_i^0 or v_i^1 for all vertical lines i of the original layout.

By scanning the path graph G from s to t, the minimum number of appropriate directed paths as well as the paths themselves can be determined in time linear in the spread of the layout.

Remark The problem can also be formulated as a shortest path problem in the *weighted path graph* $G_{WP} = (V, E_{WP}; w)$, where again V corresponds to the possible states of the vertical lines, but for all i, $(v_i^0, v_{i+1}^0), (v_i^0, v_{i+1}^1), (v_i^1, v_{i+1}^0), (v_i^1, v_{i+1}^1) \in E_{WP}$, as well as the edges from s and to

t. The *weight* w is defined as $w(e) := 0$ for all $e \in E_{WP} \cap E_P$, and $w(e) := 1$ for $e \in E_{WP} \backslash E_P$. Then any shortest s-t-path (with respect to w) in G_{WP} is equivalent to a minimum stretching to guarantee a V-partition of the corresponding layout.

In case the core consists of different components or is not V-convex, the main problem is that for a minimum stretching different components respectively two parts of the same component that are horizontally separated by a hole must not be considered independently of each other. The following lemma helps to overcome this problem.

Lemma 7 *Consider m consecutive vertical grid lines in a V-convex layout core that do not admit a V-partition. If there are columns $\vec{i}$ and $\vec{j}, 1 \leq i < j < m$, such that the layout part between 1 and m stretched along $\vec{i}$ or stretched along $\vec{j}$ admits a V-partition, then for any column $\vec{k}, i \leq k \leq j$, the layout stretched along $\vec{k}$ admits a V-partition.*

Proof

Look at the path graph of the layout. There is a directed path from some v_1-vertex to v_j^0 or v_j^1, and a directed path from v_i^0 or v_i^1 to some v_m-vertex. These paths are possibly vertex-disjoint, but overlap in the interval $[i, j]$ in the sense that both of them contain a v_k-vertex for any $k \in [i, j]$. Now, a stretching of the layout along an arbitrary column $\vec{k}$ between $\vec{i}$ and $\vec{j}$ transforms the path graph to a new path graph where these two paths are connected, and induce an s-t-path in the new path graph.

In the path graph of a V-convex layout core of spread n, for any $i, 1 \leq i \leq n$, there exist at most two disjoint paths overlapping at i. We can easily determine all maximal intervals where two disjoint paths overlap. These intervals are pairwise disjoint. Then, with lemma 3.4, any choice of a set $S \subseteq \{1, \ldots, n\}$ containing exactly one element of each of these intervals induces a minimum stretching to admit a V-partition. Using this fact, we can formulate the problem for a *single* layout core consisting of V-convex components directly as a minimum *clique cover problem* in *interval graphs*, which is solvable in time $\mathcal{O}(|V| + |E|)$ [6]. In case the interval representation is given, the problem can be solved even in time $\mathcal{O}(|V|)$.

Consider a layout core that consists of V-convex core components $C_1, \ldots, C_r$. For each of these C_i consider the path graph $G_p(C_i)$. Then the maximal intervals of overlapping disjoint paths for each of the $G_p(C_i)$ canonically forms an interval graph. The intervals are the vertices of the graph and two vertices are connected by an undirected edge iff the corresponding intervals are not disjoint.

The algorithm solving the minimum clique cover problem for interval graphs, assumed the interval representation is known, can be described as follows. Traverse the interval representation from left to right. Whenever a left endpoint of an interval is arrived this interval is labeled. When a right endpoint of an labeled interval is reached, the clique induced by this point is chosen as the covering clique, and all labeled intervals are deleted. This proceeding may be viewed as always taking the last chance, i.e. the rightmost covering clique.

For a layout containing core components of arbitrary shape the case may occur that the edges of a vertical gridline that are covered by the same core component form several disjoint vertical intervals. Then the problem is somewhat more complex at columns $\vec{i}$ for which a non-V-convex core component induces a combination of vertical intervals on gridline i and $i+1$ with the following properties. The edges of i covered by the core component form disjoint vertical intervals $I_1, \ldots, I_s$, and edges of $i+1$ covered by this core component form an interval J such that J shares some track with at least two of the I_j. We call such a column $\vec{i}$ a *critical* column. If we partition a non-V-convex core component into maximal V-convex subcomponents, say of rectangle shape (as shown in figure 6), a column is critical only if it is the boundary between two or more V-convex subcomponents. For columns that are not critical, the problem is solved by applying the usual minimum clique cover algorithm.

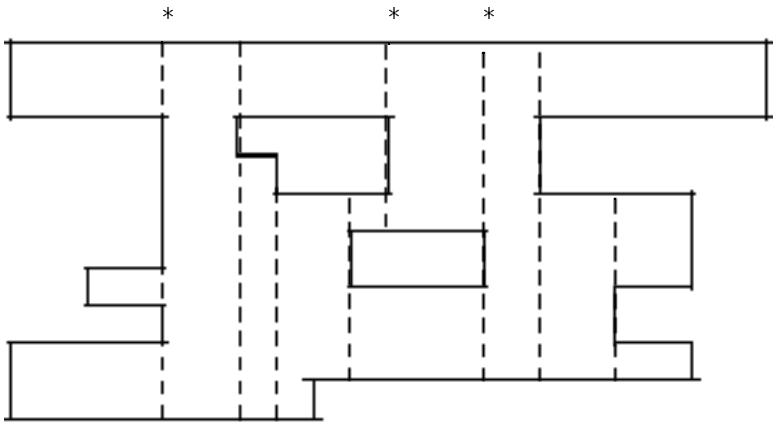

Figure 6: A partition of a non-convex core component into rectangles. The columns marked with * are critical.

So, consider a critical column $\vec{i}$. Let $I_r, \ldots, I_t$ be those intervals of gridline i which share some track with the same interval J of $i+1$. For a V-partition, the choices of appropriate states (i.e. of vertical partition edges) is for all I_j indepent of each other, if we do not consider J. Indeed, for the choice of the state combination of the I_j and J, the existence of a legal combination for each of those pairs does not suffice. But the existence of a *one* choice of state for J that admits a legal combination for *all* these pairs *simultaneously* must be guaranteed. Obviously, at most one additional stretching, namely along column $\vec{i}$, is necessary. Since, in case none of the two possible states of J admits a legal state combination for all the I_j simultaneously, a stretching along $\vec{i}$ induces that all the I_j and J belong to two different core components.

The usual minimum clique cover algorithm for interval graphs may be applied in order to solve this problem. During the proceeding of the algorithm, we only have to care of points corresponding to critical columns. When the algorithm arrives at critical column $\vec{i}$, it has to check if the clique cover determined for the intervals to the left of $i+1$ induce a correct state combination for all pairs I_j and J, using the same state of J. If this is not the case, the clique covering interval $[i]$ has to be added to the clique cover, which just induces a stretching along $\vec{i}$. Otherwise, the algorithm may proceed in the usual way. This means, that the decision if an interval $[i]$ corresponding to a critical column $\vec{i}$ has to be considered for the minimum clique cover problem is made on-line.

Clearly, the interval graph corresponding to a general layout can be determined in advance as well. Consider a core-component that is not V-convex, and a partition into V-convex subcomponents. Additionally to the intervals induced by the path graphs of the V-convex subcomponents, we have to determine appropriate intervals corresponding to the critical columns. So, let $\vec{i}$ be a critical column and $I_r, \ldots, I_t$ resp. J the vertical intervals to be considered. Each I_j, as well as J, belongs to a different V-convex subcomponent. Consider the corresponding path graphs resp. the vertices v^0_{i+1} and v^1_{i+1} corresponding to J. For each I_j we have to consider the maximal paths terminating with v^0_{i+1} resp. v^1_{i+1} in the corresponding path graph. If for an I_j there exists a path, say from a vertex corresponding to gridline l terminating e.g. with v^0_{i+1}, any stretching in the interval $[l-1, i]$ yields a path from the source of the path graph to v^0_{i+1}, assumed there exists such a path to v^1_{i+1}. So, for all the I_j we have to determine the intersection of all intervals induced by a maximal path terminating with v^0_{i+1} resp. v^1_{i+1}. Obviously, in case for one I_j there exists no path terminating with v^0_{i+1} resp. v^1_{i+1}, the interval $[i]$ is chosen.

Then, the union of these two intervals is the additional interval to be considered for the minimum clique cover problem. For assume we stretch along some column from the intersection of all intervals corresponding to paths which terminate with v^0_{i+1} resp. v^1_{i+1}. Then a legal state combination where

the state of J is the same for all pairs I_j and J is guaranteed.

It suffices to take the union of the two intervals, because only for one state of J all those legal state combinations must exist.

Observe that the union and intersection here corresponds to the $\vee$ and $\wedge$ in the 2SAT formulation in section 3.1.

Altogether, we have the following result.

Theorem 8 *A minimum stretching to admit a V-partition is equivalent to a minimum clique cover of the corresponding interval graph.*

Appendix

We have implemented the methods described in section 3. Applying this approach, examples of layouts constructed by the algorithms from [15] and [10] have been found where layout modifications are really necessary to obtain a legal three-layer partition by adding only vertical partition edges. It is proved, that there exist layouts containing only two knock-knees per column, e.g. computed by the algorithm from [17], that admit no such legal partition. But these examples are fairly large.

Experiments show that in most cases three-layer wirings exist for layouts containing only a small number of knock-knees, as e.g. those guaranteed by the algorithms from [17] and [13]. Even for layouts from [5], [4], where the number of knock-knees performed by a single net is more than a constant, only a small number of stretchings is necessary to derive three-layer wirable stretched layouts.

Figure 7: The method applied to layouts computed by the rectangle routing algorithm of Mehlhorn and Preparata

Figure 8: The method applied to a layout computed by the algorithm of Frank.

Figure 9: The method applied to a layout computed by the wirelength optimal algorithm of Formann, Wagner and Wagner.

References

[1] M.L. Brady, D.J. Brown. VLSI routing: Four layers suffice. In: *Advances in Computing Research 2 (VLSI Theory) (ed. F.P. Preparata) JAI Press* (1984) 245-257

[2] M.L. Brady, M. Sarrafzadeh. Stretching a knock-knee layout for multilayer wiring. *IEEE Trans. Comput. 39* (1990) 148-152

[3] S. Even, A. Itai, A. Shamir. On the complexity of time table and multicommodity flow problems. *SIAM J. Comput. 5* (1976) 691-703

[4] M. Formann, D. Wagner, F. Wagner. Routing through a dense channel with minimum total wire length. *Proc. of the Second Ann. ACM-SIAM Symposium on Discrete Algorithms* (1991) 475-482

[5] A. Frank. Disjoint paths in a rectilinear grid. *Combinatorica 2* (1982) 361-371

[6] M. C. Golumbic. Algorithmic Graph Theory and Perfect Graphs. *Academic Press* (1980)

[7] T. Gonzales, S. Zheng. Simple Three-layer channel routing algorithms. *Proc. of Aegean Workshop on Computing LNCS 319 (ed. J.H. Reif)* (1988) 237-246

[8] T. Gonzales, S. Zheng. On ensuring three-layer wirability by stretching planar layouts. *INTEGRATION: The VLSI Journal 8* (1989) 111-141

[9] M. Kaufmann, P. Molitor. Minimal stretching of a layout to ensure 2-layer wirability. *to appear in INTEGRATION: The VLSI Journal*

[10] R. Kuchem, D. Wagner, F. Wagner. Area-optimal three-layer channel routing. *Proc. of the 30^{th} Ann. Symposium on Foundations of Computer Science* (1989) 506-511

[11] W. Lipski, Jr. On the structure of three-layer wirable layouts. In: *Advances in Computing Research 2 (VLSI Theory) (ed. F.P. Preparata) JAI Press* (1984) 231-243

[12] W. Lipski, Jr., F.P. Preparata. A unified approach to layout wirability. *Mathematical Systems Theory 19* (1987) 189-203

[13] K. Mehlhorn, F.P. Preparata. Routing through a rectangle. *J. ACM 33* (1986) 60-85

[14] K. Mehlhorn, F. P. Preparata, M. Sarrafzadeh. Channel routing in knock-knee mode: Simplified algorithms and proofs. *Algorithmica 1* (1986) 213-221

[15] F.P. Preparata, W. Lipski, Jr. Optimal three-layer channel routing. *IEEE Trans. on Computers 33* (1984) 427-437

[16] I. G. Tollis. A new algorithm for wiring layouts. *Proc. of the Aegean Workshop on Computing LNCS 319 (ed. J.H. Reif)* (1988) 257-267

[17] D. Wagner. A new approach to knock-knee channel routing. *Proc. of the International Symposium on Algorithms LNCS 557 (eds. W. L. Hsu, R. C. T. Lee)* (1991) 83-93

[18] C. Wieners-Lummer. Three-layer channel routing in knock-knee mode. Preprint Universität Paderborn