Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

DIPLOMOVÁ PRÁCE

Jan Štola

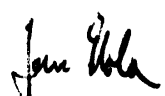
Chromatic invariants in graph drawing

Katedra Aplikované Matematiky
Vedoucí diplomové práce: Prof. RNDr. Jan Kratochvíl, CSc.
Studijní program: Informatika, Teoretická informatika

Na tomto místě bych rád poděkoval svému vedoucímu diplomové práce Prof. RNDr. Janu Kratochvílovi, CSc. za cenná doporučení a připomínky, které mi po celou dobu vzniku této práce uděloval.

Prohlašuji, že jsem svou diplomovou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce.

V Praze dne 16.4.2006



Jan Štola

Contents

Název práce: Chromatické invarianty v kreslení grafů
Autor: Jan Štola
Katedra: Katedra Aplikované Matematiky
Vedoucí diplomové práce: Prof. RNDr. Jan Kratochvíl, CSc.
e-mail vedoucího: honza@kam.mff.cuni.cz
Abstrakt: V této práci se zabýváme vlivem barevnosti grafu na existenci různých druhů ortogonálních nakreslení tohoto grafu. Studujeme otázku, jak velké můžeme volit $k_{b,n}$ tak, aby každý graf barevnosti nejvýše $k_{b,n}$ měl n-rozměrné ortogonální nakreslení s hranami s nejvýše b ohyby. $k_{b,n}$ nazýváme multipartitním číslem reprezentace/nakreslení.

Pro ortogonální nakreslení, v nichž jsou vrcholy reprezentovány úsečkami v $\mathbb{R}^n$, je v práci multipartitní číslo odvozeno přesně pro všechna n. Přesná hodnota je určena taktéž pro viditelnostní reprezentace pomocí obdélníků a čtverců. Navíc jsou vylepšeny nejlepší známé horní a dolní odhady pro trojrozměrné ortogonální nakreslení pomocí obdélníků a hranolů. Dolní odhad je zvýšen ze 3 na 22 a horní snížen ze 183 na 42.
Klíčová slova: ortogonální nakreslení grafu, barevnost

Title: Chromatic invariants in graph drawing
Author: Jan Štola
Department: Department of Applied Mathematics
Supervisor: Prof. RNDr. Jan Kratochvíl, CSc.
Supervisor's e-mail address: honza@kam.mff.cuni.cz
Abstract: This paper studies the question: What is the maximum integer $k_{b,n}$ such that every $k_{b,n}$-colorable graph has a b-bend n-dimensional orthogonal box drawing?

We give an exact answer for the orthogonal line drawing in all dimensions and for the 3-dimensional rectangle visibility representation. We present an upper and lower bound for the 3-dimensional orthogonal drawing by rectangles and general boxes. Particularly, we improve the best known upper bound for the 3-dimensional orthogonal box drawing from 183 to 42 and the lower bound from 3 to 22.
Keywords: orthogonal graph drawing, colorability

1 Introduction

The visualization of relational information has many applications in various domains. The domain entities are usually modeled as vertices and the relationships among entities are represented by edges.

There has been many graph drawing styles studied in the literature. In this thesis we study the orthogonal box drawing. This drawing has received a wide attention recently due to its applications: 2-dimensional variants in VLSI routing, circuit board layout, CASE tools etc. and 3-dimensional variants for example in packaging algorithms [1,5,6,7,8].

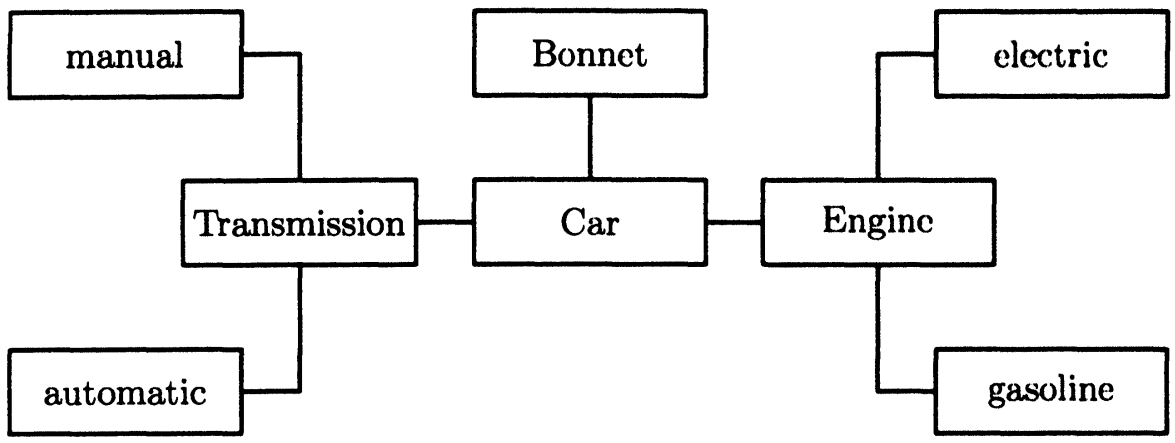


Figure 1: Many UML diagrams are orthogonal box drawings

The orthogonal box drawing represents vertices by axis-parallel boxes. Edges are drawn as axis-parallel polylines with ends on boundaries of connected boxes. The edges don't intersect another boxes and with the exception of the 2-dimensional drawing an edge cannot intersect another edge.

We call the drawing b-bend if each edge consists of at most $b+1$ line segments. A 0-bend drawing is called a straight-line drawing. If all edges in a straight-line box drawing in $\mathbb{R}^3$ are parallel then we can ignore the thickness of the boxes in this direction. We obtain a representation known as a 3D rectangle visibility drawing.

It turns out that the recognition of graphs with the given type of orthogonal drawing is difficult. For example, Shermer [11] shows that the recognition of graphs with 2-dimensional straight-line orthogonal drawing is NP-complete. Fekete et al. [12] establishes NP-completeness of recognition of graphs with a 3D rectangle visibility drawing by squares.

If we cannot effectively decide whether a graph has a drawing of the given type then it is natural to look for classes of graphs for which this decision is possible. The previous research was concentrated mainly on the complete graphs e.g. on the determination of the maximum size of a complete graph

with a drawing [2,3,4]. Unfortunately such results don't tell us much about drawing of graphs with more vertices.

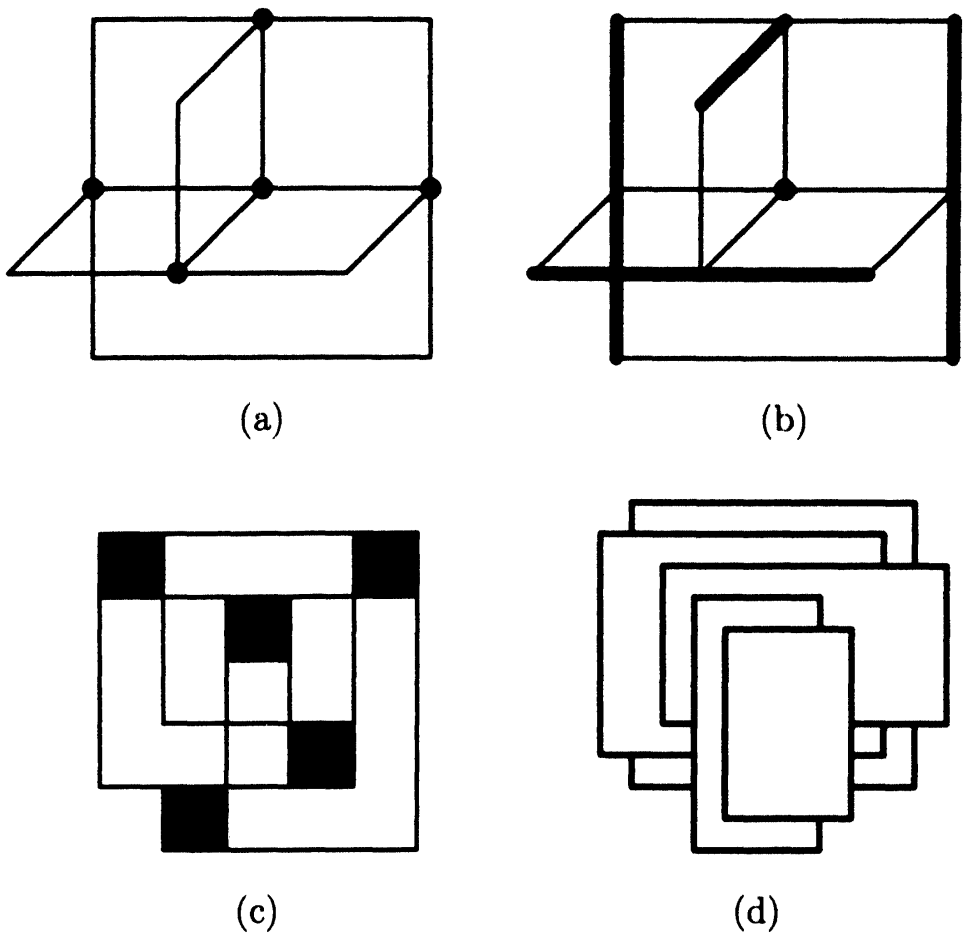


Figure 2: Different drawings of K_5:
(a) 2-bend 3-dimensional orthogonal point drawing
(b) straight-line 3-dimensional orthogonal line drawing
(c) 2-bend 2-dimensional orthogonal square drawing
(d) 3-dimensional rectangle visibility drawing

Our search for a more practical class of graphs has been inspired by the open problem presented by Wood [1]:

> What is the maximum $k \in \mathbb{Z}^+$ such that every k-colorable graph has a straight-line 3D orthogonal box drawing?

We study graphs with bounded colorability in this thesis. Every k-colorable graph is a subgraph of a k-partite graph that is itself k-colorable. Therefore it is sufficient to study the drawing of k-partite graphs.

Definition: The multipartite number of the given type of drawing is the maximum $k \in \mathbb{N}$ such that every k-partite graph has a drawing of that type. We say that the multipartite number is infinite when every multipartite graph has such a drawing.

Wood [1] proves that the multipartite number of the straight-line orthogonal box drawing is at least 3. On the other hand Fekete and Meijer [2] shows that it is at most 183.

We improve the lower bound from 3 to 22 and the upper bound from 183 to 42. We also determine the exact value of the multipartite number of the orthogonal drawing by line segments and of the rectangle visibility drawing. The Table 1. summarizes the results presented in this work.

v	d	b	multipartite number	
1	1	0	1	Theorem 1.
	2	≥ 2	∞	Section 3.3
		1	2	Theorem 2.
		0	1	Theorem 1.
	3	≥ 1	∞	Theorem 2.
	≥ 3	0	3	Theorem 1.
2	2	≥ 1	∞	Section 4.4
		0	1	Section 4.3
	3	0	$\in \langle 22, 42 \rangle$	Theorems 5., 6.
3	3	≥ 1	∞	Section 5.2
		0	$\in \langle 22, 42 \rangle$	Theorems 5., 6.
rectangle visibility drawing			8	Theorem 3.
square visibility drawing			6	Theorem 4.

Table 1. Multipartite number of d-dimensional b-bend orthogonal drawing by v-dimensional boxes.

2 Preliminaries

First of all we recall the exact definition of terms mentioned in the introduction.

Definition: d-dimensional b-bend orthogonal drawing of a graph G is a graph representation where

- $v \in V(G)$ is represented by an axis-aligned box $B_v \subseteq \mathbb{R}^d$
- $\forall v, w \in V(G), v \neq w \Rightarrow B_v \cap B_w = \emptyset$
- $(v, w) \in E(G)$ is represented by an axis-aligned polyline (with $b + 1$ line segments) connecting points on the surface of B_v and B_w
- edges (with the exception of their endpoints) don't intersect vertices
- if $d > 2$ then the edges don't intersect other edges
- if $d = 2$ then the edges can have a finite number of intersections ■

Many times we impose additional constraints on the shape of vertices. Even the rectangle visibility drawing can be defined as a special case of the orthogonal drawing.

Definition: A rectangle visibility drawing of a graph is a 3-dimensional 0-bend orthogonal drawing where

- vertices are represented by rectangles placed in parallel planes
- edges are orthogonal to the planes of rectangles ■

Usually we suppose that the rectangles are parallel to xy-plane and the edges are parallel to z-axis. An example of a rectangle visibility graph can be found on the Figure 2d.

2.1 Bar-visibility Graphs

The bar-visibility drawing is a 2-dimensional variant of the rectangle visibility drawing. It appears frequently during the study of the orthogonal drawing, especially in straight-line line drawing.

Definition: A graph G is a bar-visibility graph if it has a drawing in a plane with the following properties:

- vertices are represented by horizontal line segments,
- each edge is a vertical line (connecting two vertices) that doesn't intersect another vertex. ■

The bar-visibility graphs are characterized by Tamassia & Tollis [9] and Wismath [10].

Theorem: [9][10] *A graph G is a bar-visibility graph if and only if it has a planar embedding with all cutvertices in the exterior face.*

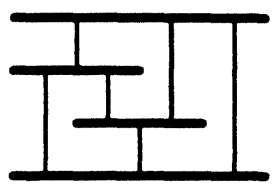


Figure 3: A bar-visibility representation of K_4

Very few bipartite graphs are bar-visibility graphs because $K_{3,3}$ is not a planar graph.

Corollary: $K_{m,n}$, $\min(m,n) \geq 3$ *is not a bar-visibility graph.*

2.2 Basic Lemmas

The next lemma is a simple application of the pigeon-hole principle. We include it because we use this formulation several times in the sequel.

Lemma 1. *Let $k, n, c \in \mathbb{N}$ and G be a complete k-partite graph whose each part has at least $c(n-1)+1$ vertices and each vertex has one of c colors. Then G contains a complete k-partite subgraph whose each part is monochromatic and contains at least n vertices.*

Proof: Each part contains at least $c(n-1)+1$ vertices. Therefore there are at least n vertices with the same color in each part. These monochromatic sets form the k-partite subgraph with the required properties. ■

We use this lemma when each vertex from a drawing must have one property from a finite set of properties and we have to ensure that the vertices from the same part have the same property. A similar situation occurs when the properties are assigned to edges.

Lemma 2. *Let $k, n, c \in \mathbb{N}$ and G be a complete k-partite graph whose each edge has one of c colors. There exists $N_{k,n,c} \in \mathbb{N}$ such that if each part of*

G has at least $N_{k,n,c}$ vertices then G contains a complete k-partite subgraph whose each part has at least n vertices and for each pair of parts the edges among elements of these parts are monochromatic.

Proof: Let N be an integer. Fix two parts B and W (call them black and white) that have at least N vertices. We prove that if $N > (n-1)c(2c)^n$ then there is a subgraph of G that is also complete k-partite, has at least n black and n white vertices and the edges among these vertices are monochromatic. The demanded subgraph can be obtained by a repeatable application of this fact.

As the first step select for each black vertex the color that appears the most frequently among its edges going to the white vertices. In the second step collect these colors and call red the one that appears the most frequently. Finally select the vertices whose the most frequent color from the step one was red. We obtain a set B' of at least N/c black vertices, each of them connected to at least N/c white vertices by red edges.

Let's count pairs $\{(v,S)|v \in B', S \subset W, |S| = n, \forall w \in S : vw \text{ is red}\}$. Each black vertex appears in at least $\binom{N/c}{n}$ pairs. Hence, there are at least $\frac{N}{c}\binom{N/c}{n}$ pairs. On the other hand if there is a set S that is in n pairs then this set with the black vertices from the pairs form the required configuration. This must happen if the number of the pairs is bigger than $(n-1)\binom{N}{n}$. But this holds because $N > (n-1)c(2c)^n$:

$$\frac{\frac{N}{c}\binom{N/c}{n}}{(n-1)\binom{N}{n}} = \frac{N}{c(n-1)} \prod_{i=0}^{n-1} \frac{N/c - i}{N-i} > \frac{N}{c(n-1)} \left(\frac{1}{2c}\right)^n > 1$$

■

Sometimes we need to separate the parts with respect to some function on the set of vertices.

Lemma 3. *Let $k, n \in \mathbb{N}$ and $G(V,E)$ be a complete k-partite graph whose each part has at least $(n-1)k+1$ vertices. For each $\ell : V \to \mathbb{R}$ there exists a complete k-partite subgraph G' of G whose each part has at least n vertices and whose parts are ℓ-separated e.g. for each pair P_1, P_2 of parts it is either $\forall x \in P_1 \forall y \in P_2\ \ell(x) \leq \ell(y)$ or $\forall x \in P_1 \forall y \in P_2\ \ell(y) \leq \ell(x)$.*

Proof: Sort the vertices $v \in V$ according to their value $\ell(v)$. Let P be the part whose n-th vertex (with respect to this order) has the lowest index in the sequence. Remove the vertices before the selected one from the sequence. Put the first n vertices of P into G' and remove the elements of P from the sequence. Continue in the same way until the sequence is empty.

Let's fix some part P. If P is not the selected part then at most $n-1$ of its vertices are removed from the sequence. Therefore at most $(n-1)(k-1)$ of

its vertices are removed before the part is selected. P has at least $(n-1)k+1$ elements. So, the described algorithm selects n vertices from each part. The resulting graph G' obviously has the required property. ■

Lemmas 1., 2. and 3. ensure the existence of a subgraph $G'(V', E')$ of a graph $G(V, E)$ such that $|V'| \geq f(|V|)$, where f is a non-decreasing function unbounded from the top. These properties of f ensure that the size of G' can be made arbitrarily big if we take the original graph G sufficiently large. We use this fact many times in the sequel because we usually want to prove the existence of a large graph G' with some properties and are not interested in the exact size of G that must be taken to find a subgraph of the required size.

3 Line Drawing

3.1 Straight-line Line Drawing

In this section we determine the multipartite number of the straight-line line drawing in the n-dimensional space e.g. the drawing where each vertex is represented by an axis-parallel line segment in $\mathbb{R}^n$.

Lemma 4. *The multipartite number of the straight-line line drawing in* $\mathbb{R}^n$ *is at most* 3 *for* $n > 2$ *and it is 1 for* $n = 1, 2$.

Proof: Let's suppose that we have a straight-line line drawing in $\mathbb{R}^n$ of a k-partite graph G. If we color the vertices according to their direction then Lemma 1. tells us that there exists a large k-partite subgraph G' with parallel vertices in the individual parts.

Now color the edges of the graph G' according to their direction. Let G'' denote the result of the application of Lemma 2. on the graph G'. The edges between arbitrary two parts of G'' are parallel.

We know that the vertices in the individual parts are parallel. We claim that the vertices from the different parts cannot be parallel. Suppose that the opposite holds e.g. there are parts P and Q such that the vertices from $P \cup Q$ are parallel (to a vector $\vec{e}_1$). The edges between P and Q are parallel (to a vector $\vec{e}_2$) due to the definition of G''. This means that P and Q together with the edges between them lie in a plane (given by vectors $\vec{e}_1$ and $\vec{e}_2$). So, we have a bar-visibility graph of $K_{|P|,|Q|}$, but the sets P and Q can be made arbitrarily large. That is in a contradiction with the planarity of bar-visibility graphs.

If P and Q are two parts of G'', P parallel to $\vec{e}_1$, Q parallel to $\vec{e}_2$ and the edges between P and Q parallel to $\vec{e}_3$ then the drawing of $K_{|P|,|Q|}$ lies in $S = p + \vec{e}_1\mathbb{R} + \vec{e}_2\mathbb{R} + \vec{e}_3\mathbb{R}$, where p is a point of some line in $P \cup Q$. If $k > 3$ then there must exist a part R parallel to vector $\vec{e}_4 \notin \vec{e}_1\mathbb{R} + \vec{e}_2\mathbb{R} + \vec{e}_3\mathbb{R}$. The edges between P and R are parallel to a vector $\vec{v}$.

Choose $l \in R$. $|l \cap S| \leq 1$ because $\vec{e}_4 \perp S$.

If $l \cap S = q$ then the edges between P and R have one end in q. There can be at most two such edges (from the directions $\vec{v}$ and $-\vec{v}$). Therefore $|P| \leq 2$, but P can be arbitrarily large.

If $l \cap S = \emptyset$ then $l + \vec{v}\mathbb{R}$ intersects S in at most one point. This means that l can be connected to at most one vertex from P and we have a contradiction with the size of P again. Hence $k \leq 3$.

The case $n = 1$ is obvious. If $n = 2$ then $G(V, E)$ is a union of two planar (bar-visibility graphs) and as such has less than $12|V|$ edges. Therefore no sufficiently large bipartite graph has a 2D straight-line line drawing. ■

The previous proof utilizes the fact that we can choose from an arbitrary complete multipartite graph a subgraph that is itself a complete multipartite graph, has some specific property and can be made arbitrarily large if the original graph is sufficiently big. The additional property allows us to simplify the proof. We use this method in many of the following proofs.

The lower bound on the multipartite number that matches our upper bound e.g. the construction of a 3-dimensional straight-line line drawing of $K_{a,b,c}$ for arbitrary positive integers a, b, c is given by Wood.

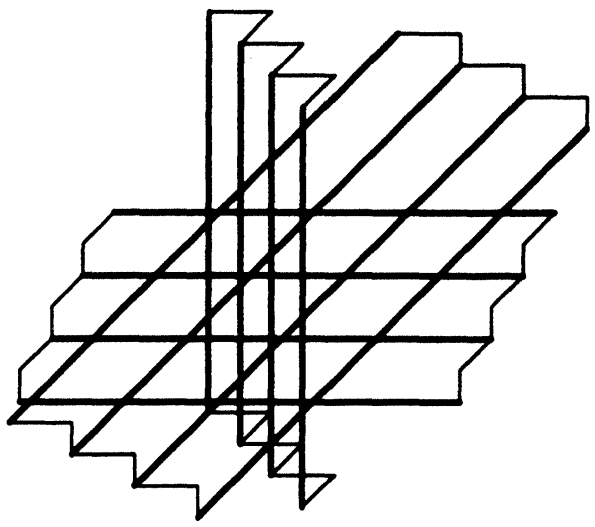


Figure 4: Straight-line 3-dimensional orthogonal line drawing of $K_{n,n,n}$

Lemma 5. [1] *The multipartite number of the straight-line line drawing in* $\mathbb{R}^3$ *is at least* 3.

Proof: It is sufficient to show that $K_{n,n,n}$ has such a drawing for each $n \in \mathbb{N}$. The graph $K_{n,n,n}$ can be represented by the following lines

$$\begin{aligned} u_i: &\quad (2, 2i+1, 2i) &\rightarrow &\quad (2n+1, 2i+1, 2i) \\ v_j: &\quad (2j, 2, 2j+1) &\rightarrow &\quad (2j, 2n+1, 2j+1) \\ w_k: &\quad (2k+1, 2k, 2) &\rightarrow &\quad (2k+1, 2k, 2n+1) \end{aligned}$$

where line segments u_i, v_j resp. w_k represent the vertices from the first, the second resp. the third part.

The edges u_iv_j, u_iw_k and v_jw_k are represented by lines

$$\begin{aligned} u_i: &\quad (2j, 2i+1, 2i) &\rightarrow &\quad (2j, 2i+1, 2j+1) &: v_j \\ u_i: &\quad (2k+1, 2i+1, 2i) &\rightarrow &\quad (2k+1, 2k, 2i) &: w_k \\ v_j: &\quad (2j, 2k, 2j+1) &\rightarrow &\quad (2k+1, 2k, 2j+1) &: w_k \end{aligned}$$

The detailed proof of the correctness of this drawing is given in [1] (Theorem 3.1). ■

Theorem 1. *The multipartite number of the straight-line line drawing in* $\mathbb{R}^n$ *is* 3 *for* $n > 2$ *and it is 1 for* $n = 1, 2$.

3.2 1-bend Line Drawing

Lemma 6. *The multipartite number of the* 1*-bend line drawing in* $\mathbb{R}^2$ *is at most 2.*

Proof: We proceed similarly to the proof of Lemma 4. Suppose that we have a 1-bend 2-dimensional line drawing of a complete k-partite graph G. Due to Lemma 1. we can assume that all vertices in the individual parts are parallel.

Let's color 0-bend edges by red and 1-bend edges by green. Lemma 2. ensures the existence of a k-partite subgraph of G with monochromatic edges between each pair of parts.

If $k \geq 3$ then there must exist two parts P and Q that have vertices with the same direction. What is the color of the edges between these parts? It cannot be red because the multipartite number of the 2-dimensional straight-line line drawing is one.

When we have a 1-bend edge between two parallel line segments then one of the ends of this edge must be a continuation of one of these segments. Assign the edge to this segment. Each segment can have at most 2 edges assigned to it. Therefore the maximum number of green edges between P and Q is $2(|P| + |Q|)$, but a sufficiently large bipartite graph has more edges. Hence k must be less than 3.■

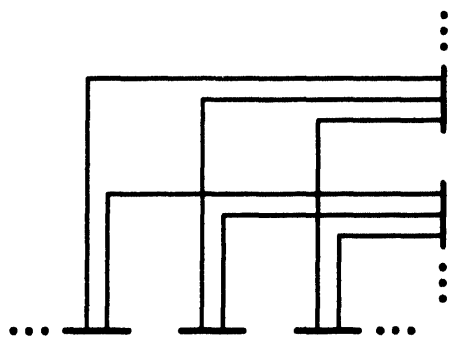


Figure 5: 1-bend 2-dimensional line drawing of $K_{a,b}$

Theorem 2. *The multipartite number of the* 1*-bend line drawing in* $\mathbb{R}^n$ *is 2 for* $n = 2$ *and is infinite for* $n > 2$.

Proof: The upper bound on the multipartite number of the 1-bend line

drawing in $\mathbb{R}^2$ is proved in Lemma 6. The lower bound is given by the construction on the Figure 5.

The multipartite number in $\mathbb{R}^n, n > 2$ is infinite because every complete graph has a 1-bend line drawing in $\mathbb{R}^3$ see Figure 6.■

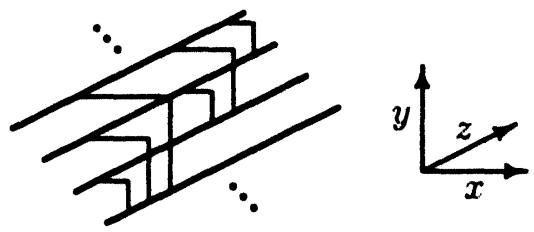


Figure 6: 1-bend 3-dimensional line drawing of K_k

3.3 2-bend Line Drawing

The Figure 7. shows that every complete graph has a 2-bend 2-dimensional line drawing. Therefore the multipartite number of this drawing is infinite.

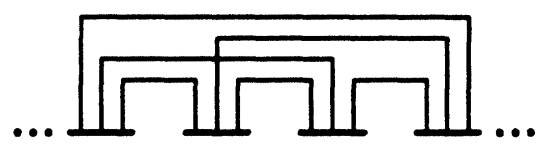


Figure 7: 2-bend 2-dimensional line drawing of K_k

4 Rectangle Drawing

In this section we work with rectangles in parallel planes as if they were in the same plane. Operations on such rectangles should be understood as operations on the projections (into one of the planes) and the projection of the result of the operation (for example the intersection of some rectangles) back into the individual planes of the rectangles.

4.1 Rectangle Visibility Drawing

Let's remind two important theorems that we use in this section.

Theorem: *(Erdős-Szekeres) Each sequence of $n^2 + 1$ distinct real numbers contains a monotonic subsequence of length at least $n + 1$.*

Theorem: *(Helly) If every at most $d + 1$ sets of a finite family of convex sets in $\mathbb{R}^d$ intersect then all the sets of the family intersect.*

The sufficient and necessary condition for the existence of a common intersection of a set of axis-aligned rectangles is even simpler.

Lemma 7. *Let S is a set of axis-aligned rectangles in a plane. If each pair of rectangles from S intersect then the rectangles in S have a common intersection.*

Proof: The rectangles in S have a common intersection if and only if their projections along x and y axis have a common intersection. The projections along some axis (e.g. some intervals) have a common intersection if and only if each two projections intersect (Helly theorem for $d = 1$). So, the rectangles in S have a common intersection if and only if projections of each pair of rectangles intersect and that happens if and only if each two rectangles intersect. ■

Definition: Let B is a box in an orthogonal drawing. $x^+(B)$ denotes the maximum coordinate of a point in the box B. Similarly we define x^-, y^+, y^-, z^+ and z^-.

Note that $z^+ = z^-$ in the rectangle visibility drawing. We denote this function simply by z there.

Lemma 8. *The multipartite number of the rectangle visibility drawing is at most 8.*

Proof: Suppose that we have a rectangle visibility drawing of a complete k-partite graph G. Apply Lemma 3. on this graph consecutively with functions

z, x^+, x^-, y^+ and y^-. We obtain a graph G' with parts separated with respect to these functions.

Take an arbitrary part P_i of G' and sort its elements according to their z-coordinates. Due to Erdős-Szekeres theorem we can choose from this sequence a subsequence P_i' of length at least $|P_i|^{1/16}$ that is monotone in x^+, x^-, y^+ and y^- coordinates. Denote by G'' the complete k-partite graph with parts P_i'. From the construction of G'' it is obvious that its parts can be made arbitrarily large if we take G with sufficiently large parts.

We claim that we can suppose that the orthogonal projections (along z-axis) of rectangles from G'' have a common intersection. Rectangles from the different parts must intersect to be able to see each other. Due to the previous lemma it is sufficient to show that each part has a common intersection. That happens if and only if each two rectangles from this part intersect.

Let P_1 be a part without a common intersection. There must be two elements $r_1, r_2 \in P_1$ that don't intersect. Without loss of generality (w.l.o.g.) it is $x^+(r_1) < x^-(r_2)$.

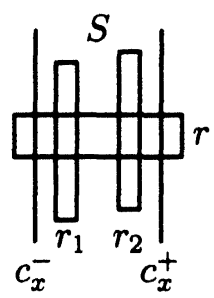


Figure 8:

G'' is a complete k-partite graph. Hence a rectangle r from a different part (to see both r_1 and r_2) must have $x^-(r) < x^+(r_1)$ and $x^+(r) > x^-(r_2)$.

Let's modify the part P_1 to have a common intersection. Let c_x^+ (respectively c_x^-) be a maximum (resp. minimum) x-coordinate of a point of a rectangle in P_1. Denote by S the y-parallel strip between c_x^- and c_x^+ (see Figure 8.)

The proved inequalities together with the x^+, x^--separability of parts ensures that $x^-(r) < c_x^-$ and $c_x^+ < x^+(r)$ for each rectangle $r \notin P_1$. Therefore if two rectangles not in P_1 can see each other through a point (x, y) in the strip S then they can see each other also through a point ($(c_x^- - \varepsilon, y)$ or $(c_x^+ + \varepsilon, y)$ for a sufficiently small $\varepsilon > 0$) outside the strip.

The rectangles in P_1 (ordered according to the z-coordinate) are monotone in x^+ and x^- coordinates and don't have a common intersection. So, they must be either increasing or decreasing in both these coordinates - the rectangles form stairs (see Figure 9.)

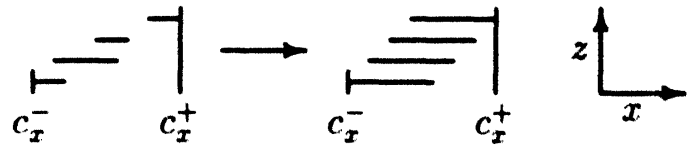


Figure 9: Modification of stairs

We claim that if we change x^+ and x^- coordinates of the stair rectangles to ensure a common intersection (as it is shown on the Figure 9.) then we don't destroy the completeness of the k-partite visibility representation of G''.

Only the rectangles in the strip S are modified. So, the visibility among rectangles not in P_1 is not affected because they can see each other through points outside the strip. It remains to show that the rectangles from P_1 can see all other rectangles.

Sides y^+ and y^- of each rectangle not in P_1 cross the whole width of the strip S. They mark on the strip (orthogonal) sub-strips. The rectangles not in P_1 can see the rectangles from P_1 only through their sub-strips.

No visibility is destroyed if we move the x^+ and x^- coordinates of rectangles from P_1 such that the same rectangles remain visible through each sub-strip, but that is exactly what our stair-modification technique does.

We have shown that we can expect each part of G'' to have a common intersection.

We know that if we sort the rectangles from some part P of G'' according to their z-coordinates then we obtain a sequence monotone also in x^+, x^-, y^+ and y^- coordinates. Moreover if P has a common intersection then we can consider P to form a frame with sides oriented up and down (see Figure 10.) The orientation determines the direction from which the corresponding sides of rectangles are visible.

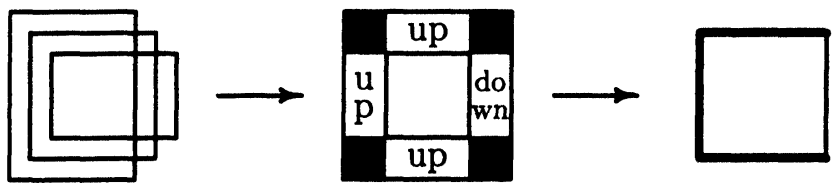


Figure 10: Transformation of one part into a frame and an oriented rectangle

We also know that the parts are x^+, x^-, y^+, y^--separated. Thus two corresponding sides of frames cannot intersect (see Figure 11.) The interiors of frames intersect because all rectangles have a common intersection. Due to these facts we can shrink the frames into rectangles with oriented sides. Now

two parts can see each other if the boundaries of their oriented rectangles intersect and the intersecting sides have a correct orientation.

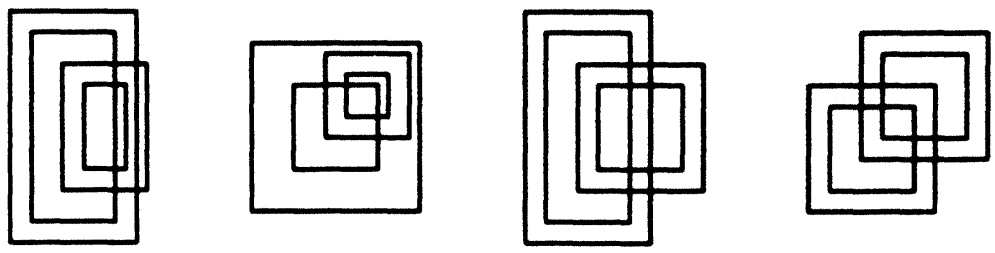


Figure 11: Examples of invalid (a), (b) and valid (c), (d) intersections of frames

It remains to prove that a complete graph with this modified oriented rectangle visibility representation has at most 8 vertices. ■

Lemma 9. *If K_n has a modified visibility representation by rectangles with oriented sides (as described in the previous proof) then $n \leq 8$.*

Proof: We proceed in a similar way as Fekete et al. [3] in the proof of the non-existence of a visibility representation of K_8 by unit squares e.g. by a computer search. Our algorithm is based on their algorithm. We modify it to generate visibility representations with general rectangles (not only squares). We also add a code that assigns an orientation to the individual sides. When the next rectangle is added the new code also checks whether the orientation requirements are satisfied.

The location of a rectangle R can be described by its z-coordinate and by the orthogonal distances of each of its sides from some (fixed) point from the common intersection of the rectangles e.g. by a 4-tuple $(x_R^+, x_R^-, y_R^+, y_R^-)$. The exact values of these coordinates are not important. It is sufficient to know the relative order of the values of the individual coordinates. Hence, we can assume that each coordinate is an integer in the range $\langle 1, n \rangle$ without changing the visibility relationships among the rectangles.

Consider two rectangles $A = (x_A^+, x_A^-, y_A^+, y_A^-)$ and $B = (x_B^+, x_B^-, y_B^+, y_B^-)$. The intersection $A \cap B$ contains the intersections of the boundaries of A and B. The corners of $A \cap B$ are therefore the only candidates through which A and B can see each other. The left top corner is an intersection of the boundaries of A and B if and only if $\min(x_A^-, x_B^-) = x_B^-$ and $\min(y_A^+, y_B^+) = y_A^+$ (or $\min(x_A^-, x_B^-) = x_A^-$ and $\min(y_A^+, y_B^+) = y_B^+$), see Figure 12. Similar conditions hold for other corners.

A and B can see each other through a corner of $A \cap B$ only if there is no rectangle R between A and B that intersects this line of visibility. For

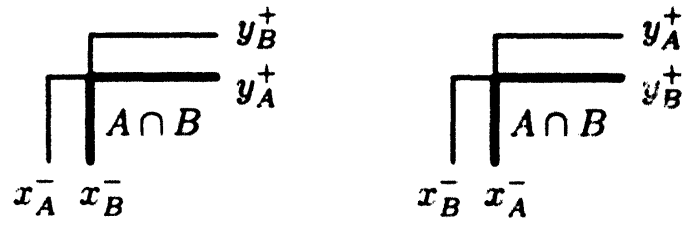


Figure 12:

example, the line of visibility corresponding to the left top corner is not intersected if and only if the condition

$$x_R^- < \min(x_A^-, x_B^-) \quad \text{or} \quad y_R^+ < \min(y_A^+, y_B^+)$$

holds for each rectangle R between A and B.

Finally if a corner of $A \cap B$ is an intersection of the boundaries of A and B and the corresponding line of visibility is not intersected then A and B are connected by an edge if and only if the intersecting edges have a correct orientation e.g. the lower one is oriented up and the upper one is oriented down.

We begin the search with one rectangle and proceed in a depth-first-search manner examining all the ways to add a new rectangle above the rectangles already added. On the m-th level of the search we have a rectangle visibility drawing of K_m described by 4-tuples with values from $\langle 1, m \rangle$. We have $m + 1$ possible choices for each coordinate for the next rectangle. For each possibility we check whether the boundary of the new rectangle intersect boundaries of other rectangles and whether the rectangles can see each other through the line of visibility given by the intersection. If this holds then we try to find the orientation of sides of the added rectangle to create a valid representation of K_{m+1}.

The following observations allow us to speed up the search:

- the lowest rectangle can have all sides oriented up,
- the second lowest rectangle can have all but one sides oriented up (one side oriented down is sufficient to see the only rectangle below).

There are only four representations of K_2 that satisfy these two rules (see Figure 13.) All other representations are reflections and/or rotations of these representations. Consider 4-tuples of the rectangles (call them R_1 and R_2) in K_2. It cannot happen that all four coordinates of R_2 are bigger (resp. smaller) than the corresponding coordinates of R_1 because otherwise the boundaries of R_1 and R_2 would not intersect. Hence, only one, two or

three coordinates of R_2 can be bigger than the coordinates of R_1. For the first and the last possibility there is only one representation of K_2 (Figure 13a and 13d). For the second possibility we can choose whether the bigger coordinates are adjacent or not (Figure 13b and 13c).

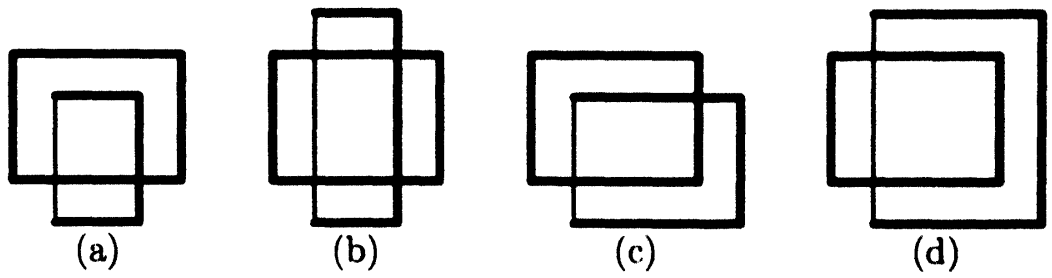


Figure 13: Modified rectangle visibility representations of K_2 (thick sides are oriented up, thin sides are oriented down)

We prove that the multipartite number of rectangle visibility drawing is at most 8. Therefore it is sufficient to show that K_9 doesn't have a modified rectangle visibility drawing. So, we can add optimizations dual to the already mentioned ones:

- the ninth rectangle can have all sides oriented down,
- the eight rectangle can have all but one sides oriented down.

These conditions also ensure that the searched space is finite. The algorithm either finds a representation of K_9 or proves that there is no such a representation.

We were able to process all valid configurations in 26 hours on Intel Centrino 1.7GHz and verified that there is no representation of K_9 with the required properties.[1] See Appendix A for the source code of the Java class used for the verification. ■

Lemma 10. *The multipartite number of the rectangle visibility drawing is at least 8.*

Proof: The Figure 14. shows (in the form of oriented rectangles) a rectangle visibility drawing of a complete 8-partite graph. The numbers in the lower right corner determine the order of parts with respect to the z-coordinate. The thick sides are oriented up, the thin sides are oriented down. The small circles show areas where the individual parts see each other. ■

[1]The algorithm was run several times on different hardware configurations in fact. Check sums were used to recognize computations affected by a potential hardware failure.

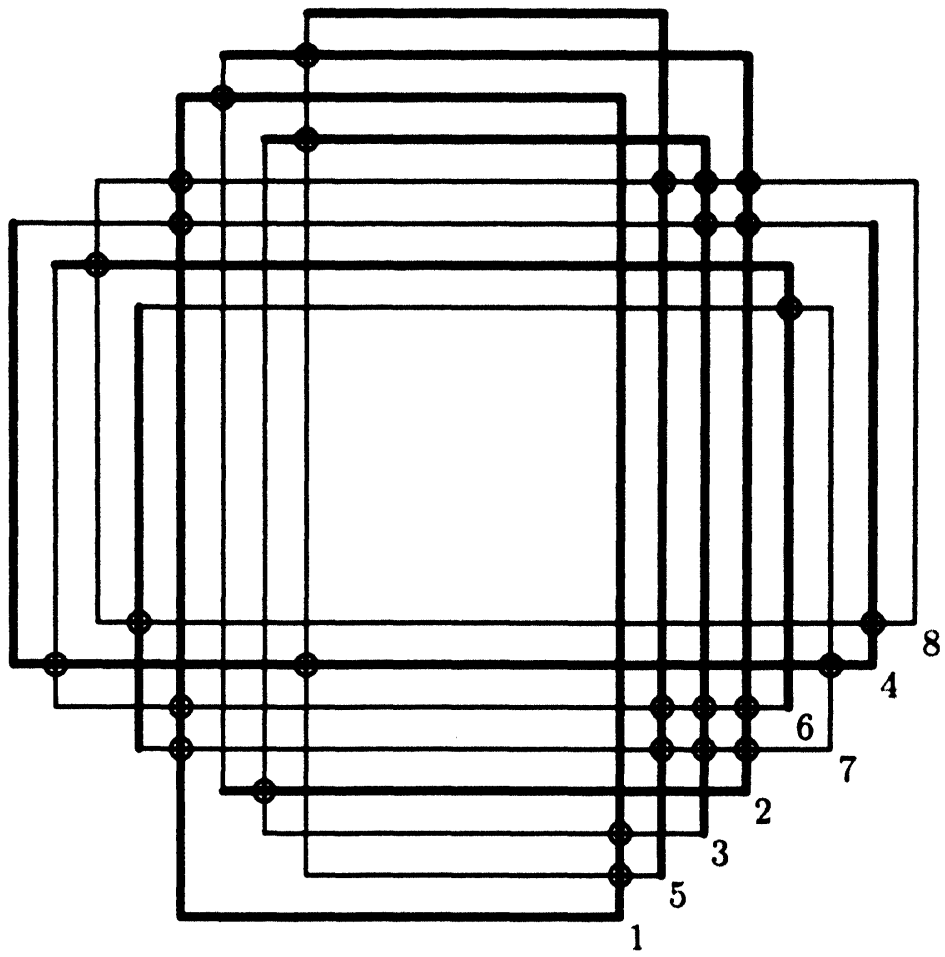


Figure 14: Rectangle visibility representation of $K_{a,b,c,d,e,f,g,h}$

If we put together Lemmas 8., 9. and 10. we obtain the following theorem.

Theorem 3. *The multipartite number of the rectangle visibility drawing is 8.*

4.2 Square Visibility Drawing

The multipartite number of the square visibility drawing (e.g. the rectangle visibility drawing where vertices are represented by unit squares) can be determined in the same way as the multipartite number of the general rectangle visibility drawing.

Lemma 11. *The multipartite number of the square visibility drawing is at most 6.*

Proof: Suppose that we have a square visibility drawing of a complete k-partite graph G. We choose a subgraph G'' of G as we do it in the proof of Lemma 8.

We claim that the squares from each part of G'' have a common intersection e.g. we don't need the stair modification technique for squares. If P_1 is a part without a common intersection then there must be two squares

$r_1, r_2 \in P_1$ that don't intersect. W.l.o.g. it is $x^+(r_1) < x^-(r_2)$. A square r from a different part (to see both r_1 and r_2) must have $x^-(r) < x^+(r_1)$ and $x^-(r_2) < x^+(r)$. The parts are x^+-separated and we know that $x^+(r_1) < x^+(r)$ therefore also $x^+(r_2) < x^+(r)$. The x^--separability of parts on the other hand gives us inequality $x^-(r) < x^-(r_1)$. Together we have

$$x^-(r) < x^-(r_1) = x^+(r_1) - 1 < x^-(r_2) - 1 = x^+(r_2) - 2 < x^+(r) - 2$$

which is in a contradiction with the equation $x^-(r) = x^+(r) - 1$ that holds because r is a unit square.

Now we can transform the parts into oriented rectangles (analogously to the proof of Lemma 8.) and assign to each rectangle R a 4-tuple $(x_R^+, x_R^-, y_R^+, y_R^-)$ with integer values from $\langle 1, k \rangle$ as we do it in Lemma 9.

The squares in G have a unit size. Therefore $x^+(r) < x^+(r')$ if and only if $x^-(r) < x^-(r')$. Hence, for two oriented rectangles A and B we have $x_A^+ < x_B^+$ if and only if $x_A^- < x_B^-$. The values x_R^+ are distinct integers from $\langle 1, k \rangle$. So, it must be $x_R^+ + x_R^- = k + 1$ for all oriented rectangles R. The analogous equations $y_R^+ + y_R^- = k + 1$ hold for y-coordinates. The oriented rectangles obtained by the transformation are therefore oriented squares.

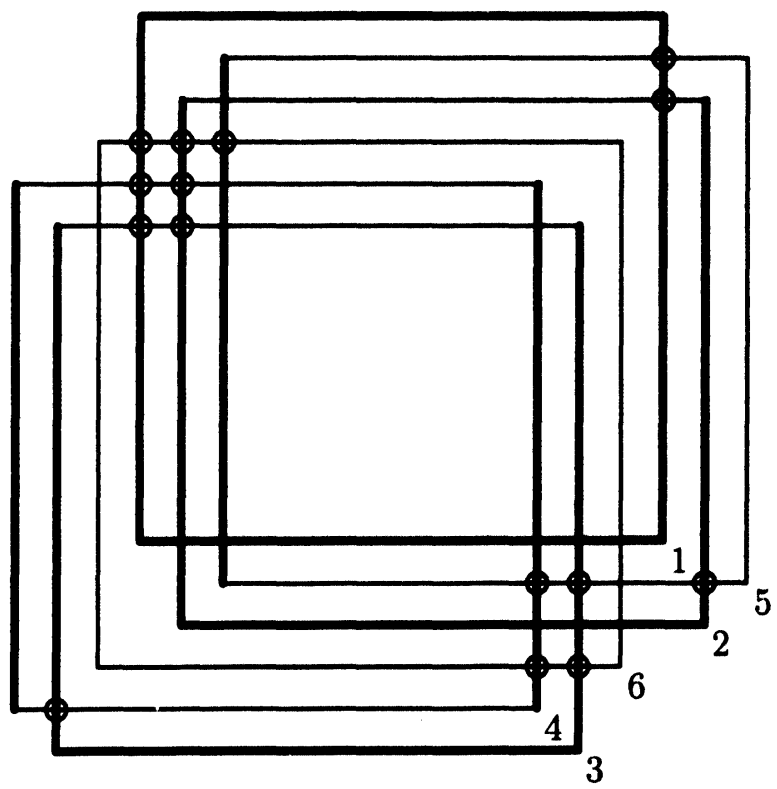


Figure 15: Square visibility representation of $K_{a,b,c,d,e,f}$

It remains to prove that a complete graph with the modified square visibility representation has at most 6 vertices. That can be verified by a slight modification of the algorithm presented in the proof of Lemma 9 – when we

add the m-th rectangle R to a drawing we have to make sure that it is a unit square e.g. that $x_R^+ + x_R^- = m + 1$ and $y_R^+ + y_R^- = m + 1$ hold for its 4-tuple $(x_R^+, x_R^-, y_R^+, y_R^-)$.

We were able to process all valid configurations in less than one second on Intel Centrino 1.7GHz and verified that there is no representation of K_7 with the required properties.■

Theorem 4. *The multipartite number of the square visibility drawing is 6.*

Proof: The Figure 15. shows (in the form of oriented squares) a square visibility drawing of a complete 6-partite graph. The matching upper bound is given by Lemma 11.■

4.3 Straight-line Rectangle Drawing

A graph with a 2-dimensional straight-line rectangle drawing is a union of two planar (bar-visibility graphs). Therefore the multipartite number of such a drawing is one.

The 3-dimensional straight-line rectangle drawing has similar properties to the 3-dimensional straight-line box drawing. The upper bound on the multipartite number of the 3-dimensional straight-line box drawing proved in the next section is also the best known bound for the straight-line rectangle drawing. On the other hand the following drawing of a complete 22-partite graph provides also the best known lower bound on the multipartite number of the 3-dimensional straight-line box drawing.

Theorem 5. *The multipartite number of the straight-line rectangle drawing in* $\mathbb{R}^3$ *is at least 22.*

Proof: The Figure 16. shows a rectangle visibility drawing of a complete 6-partite graph. Take a copy of the construction from the Figure 14. and rotate it to be parallel to xz-plane. Place the copy between the third and the fourth part. Ensure that x^+ (resp. x^-) coordinates of the rectangles from the copy are bigger (resp. smaller) than the coordinates of the rectangles from the 6-partite graph.

Take another copy of the construction from the Figure 14. and rotate it to be parallel to yz-plane. Place the copy again between the third and the fourth part either below or over the first copy. Ensure that y^+ (resp. y^-) coordinates of the rectangles from this copy are bigger (resp. smaller) than the coordinates of the rectangles from the 6-partite graph.

The copies are schematically displayed on the Figure 16. by the horizontal and the vertical line. The small circles show the areas where the individual

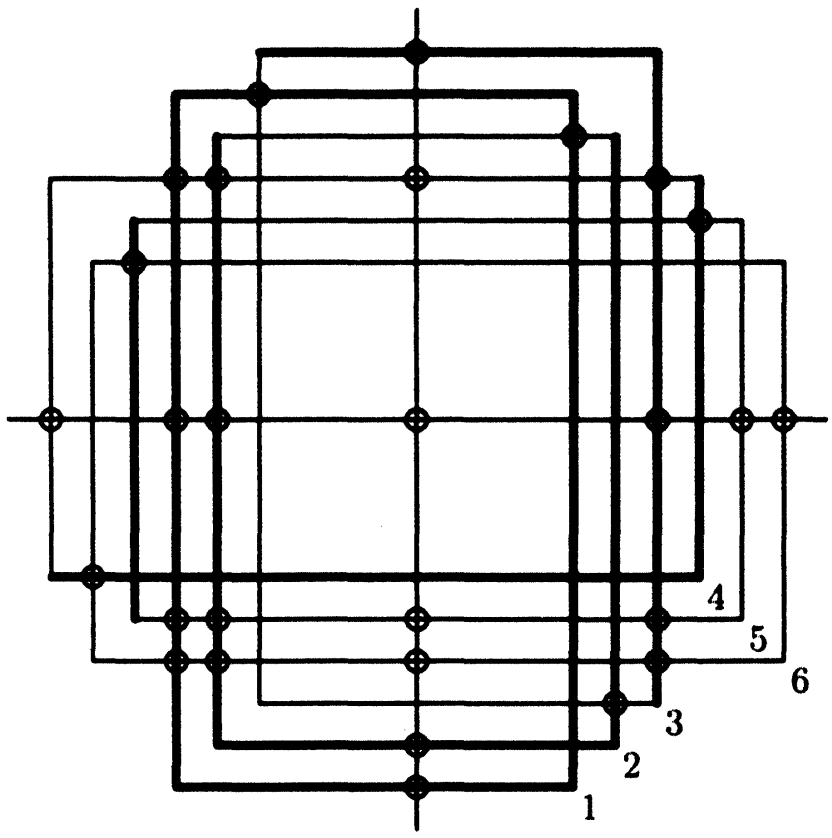


Figure 16: Straight-line rectangle drawing of a complete 22-partite graph

parts see each other. It can be easily verified that the resulting construction is a straight-line rectangle drawing of a complete 22-partite graph. ■

4.4 1-bend Rectangle Drawing

If one bend is allowed in a rectangle drawing in $\mathbb{R}^2$ then the multipartite number become infinite, see Figure 17.

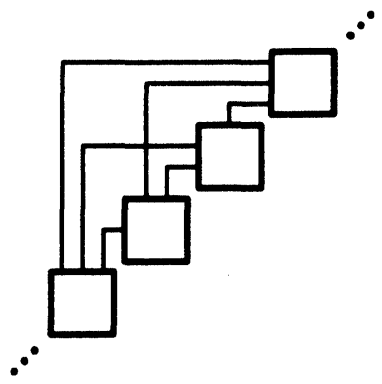


Figure 17: 1-bend 2-dimensional rectangle drawing of K_k

5 Box Drawing

5.1 Straight-line Box Drawing

Let's remind one definition from [2].

Definition: Let A and B be axis-parallel boxes in $\mathbb{R}^3$. We write $A \prec_x B$ if A and B can see each other and $x^+(A) < x^-(B)$. Similarly we define $A \prec_y B$ and $A \prec_z B$.

It can be easily verified that these relations are partial orders on the boxes in a straight-line box drawing of a complete graph. We say that some boxes from a drawing form an x-chain (resp. x-antichain) if they form a chain (resp. antichain) in the partial order $\prec_x$.

Fekete and Meijer [2] show the following properties of these orders.

Lemma 12. *Let C be a maximum length x-chain in a 3-dimensional orthogonal box drawing of a complete graph. There cannot be an x-chain D of length greater than 4 such that $C \cap D = \emptyset$.*

Proof: See Lemma 11. in [2].■

Lemma 13. *If there is no chain (x-chain, y-chain or z-chain) longer than 4 in a 3-dimensional orthogonal box drawing of a complete graph G then G can have at most 18 vertices.*

Proof: See Lemma 15. in [2].■

Lemmas 12. and 13. allow us to estimate the maximum size of a complete graph with a drawing with the bounded length of chains.

Lemma 14. *If k is the maximum length of a chain that appears in the partial order $\prec_x$, $\prec_y$ or $\prec_z$ in a 3-dimensional straight-line box drawing of K_n then $n \leq 3k + 18$.*

Proof: Let C_x, C_y resp. C_z denotes the maximum x-chain, y-chain resp. z-chain in the drawing. If we remove the boxes in $C_x \cup C_y \cup C_z$ from the drawing then by Lemma 12. there cannot remain a chain of length 5. There remain at most 18 boxes by Lemma 13. Hence, $n \leq |C_x| + |C_y| + |C_z| + 18 \leq 3k + 18$.■

The presented properties of the partial orders $\prec_x$, $\prec_y$ and $\prec_z$ can be utilized to prove an upper bound for the multipartite number of a box drawing.

Theorem 6. *The multipartite number of the straight-line box drawing in $\mathbb{R}^3$ is at most 42.*

Proof: Let G be a large complete k-partite graph that has a 3D straight-line box drawing. Color its edges by three colors according to their direction. Apply Lemma 2. on G and denote by G' the resulting graph.

Select one box from each part of G'. These boxes form a 3D straight-line box drawing of K_k. We claim that this drawing doesn't contain a chain of length greater than 8. If we prove this then by the previous lemma $k \leq 3.8 + 18 = 42$.

Suppose by contradiction that there exists (w.l.o.g.) an x-chain of length 9. The boxes from the parts with a member in this chain also (due to the selection of G') see each other along the x-axis. Therefore they correspond to a rectangle visibility representation of a 9-partite graph that can be made arbitrarily large if G is taken sufficiently large. This is in a contradiction with Theorem 3.■

5.2 1-bend Box Drawing

The infinity of the multipartite number of the 1-bend box drawing comes immediately from the infinity for the 1-bend 3D line drawing.

6 Related Results

In the previous sections we estimate the maximum k such that each k-colorable graph has a drawing of the given type. We can ask a dual question: what is the minimum k such that no k-colorable graph has a drawing of the given type? There are no non-trivial upper bounds known for this problem. The best known lower bounds are given by the drawings of complete graphs. The summary of these results is given in the Table 2.

v	d	b	size
1	1	0	2
	2	≥ 2	∞
		1	≥ 6
		0	6
	3	≥ 1	∞
	3	0	≥ 8
2	2	≥ 1	∞
		0	8
	3	0	$\in \langle 56, 183 \rangle$
3	3	≥ 1	∞
		0	$\in \langle 56, 183 \rangle$
rectangle visibility drawing			$\in \langle 22, 55 \rangle$
square visibility drawing			7

Table 2. Size of the largest complete graph with d-dimensional b-bend orthogonal drawing by v-dimensional boxes.

Lemma 15. shows that K_6 on the Figure 18a. is the largest complete graph with the 2-dimensional orthogonal straight-line line drawing. We conjecture that K_8 on the Figure 18b. has the same property for 3-dimensional orthogonal straight-line line drawing.

Lemma 15. *K_7 doesn't have a 2-dimensional straight-line line drawing.*

Proof: Suppose that we have a 2-dimensional straight-line line drawing of $K_n, n \geq 7$. W.l.o.g. there are at least as many horizontal lines as the vertical ones e.g. there are at least 4 horizontal lines.

There cannot be three horizontal lines with the same y-coordinate because the middle line would block the visibility between the other two lines.

If there are two horizontal lines l_1, l_2 with the same y-coordinate then w.l.o.g. $x^+(l_1) < x^-(l_2)$. A horizontal line l, $l \neq l_1, l_2$ must have $x^-(l) \leq x^+(l_1)$ and $x^-(l_2) \leq x^+(l)$. Therefore there cannot be another pair of horizontal lines with the same y-coordinate. If there are two horizontal lines with

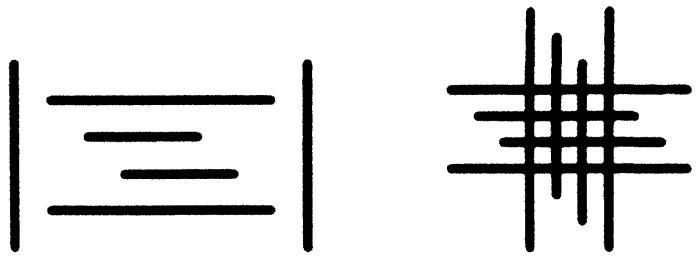


Figure 18: (a) 2-dimensional straight-line line drawing of K_6
(b) 3-dimensional straight-line line drawing of K_8

the same y-coordinate then there cannot be a vertical line. The vertical line can see both horizontal lines only if it is between them, but then it blocks the visibility between the horizontal lines.

Choose a horizontal line h that neither has the highest nor the lowest y-coordinate. The edge between a horizontal and a vertical line must be a continuation of one of the lines. The edge between h and a vertical line cannot be a continuation of the vertical line because otherwise the edge between the vertical line and the horizontal line with the highest or the lowest y-coordinate would have to intersect h. So, the edge between h and a vertical line must be a continuation of h. Hence, there are at most two vertical lines.

If there is no vertical line then there are at most $3n - 6$ vertical edges (they form a bar-visibility drawing) and at most one horizontal edge between lines with the same y-coordinate. The complete graph with n vertices has $n(n-1)/2$ edges. So, it must be $3n - 5 \geq n(n-1)/2$ and therefore $n \leq 5$.

If there is one vertical line then there are at most $3(n-1) - 6$ edges among the horizontal lines and $n - 1$ edges between the vertical line and the

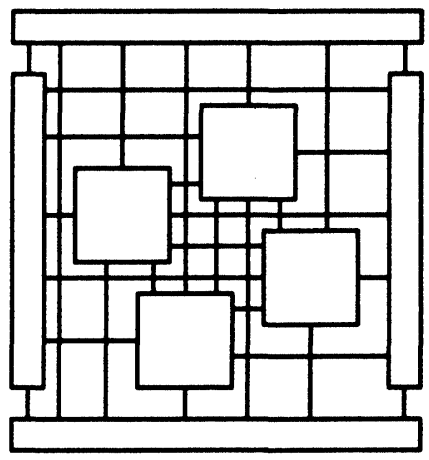


Figure 19: 2-dimensional straight-line rectangle drawing of K_8

horizontal lines. So, it must be $4(n-1)-6 \geq n(n-1)/2$ and therefore $n \leq 5$.

Finally, if there are two vertical lines then there are at most $3(n-2)-6$ edges among the horizontal lines, $2(n-2)$ edges between the vertical and the horizontal lines and one edge between the vertical lines. So, it must be $5(n-2)-5 \geq n(n-2)/2$ and therefore $n \leq 6$.■

Beineke [13] proves that K_9 is not a union of two planar graphs. If a graph has a 2-dimensional orthogonal straight-line drawing then it must be a union of two bar-visibility (e.g. planar) graphs. Therefore K_8 on the Figure 19. is the largest complete graph with this drawing.

The best known results for the visibility drawing comes from Fekete et al. [3]. K_7 has a square visibility drawing (see Figure 20.) while K_8 doesn't. The Table 3. describes the rectangle visibility drawing of K_{22}, the largest known complete graph with this drawing. K_{56} on the other hand doesn't have such a drawing.

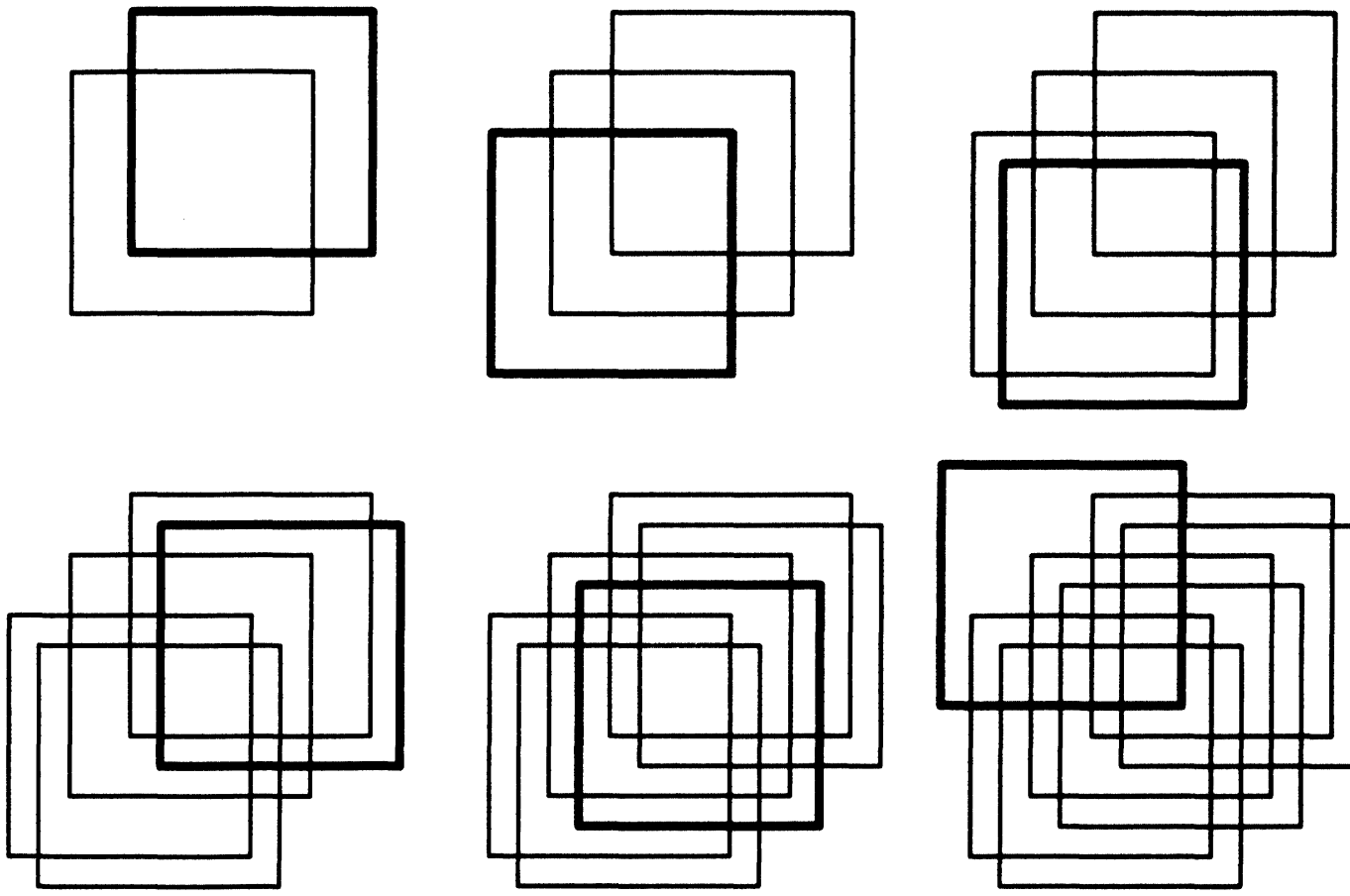


Figure 20: Rectangle visibility representation of K_7 by unit squares

Fekete and Meijer [2] use the rectangle visibility drawing of K_{22} in their construction of the 3-dimensional orthogonal box drawing of K_{56} and they show that K_{184} doesn't have this drawing.

z	1	2	3	4	5	6	7	8	9	10	11
y^+	22	11	9	8	7	5	1	17	16	4	15
y^-	22	13	6	2	19	17	18	14	12	20	15
x^+	22	15	18	12	9	8	7	5	4	1	3
x^-	22	16	15	20	8	11	12	1	2	14	3

z	12	13	14	15	16	17	18	19	20	21	22
y^+	19	14	6	3	2	20	13	10	18	12	21
y^-	16	1	3	4	5	7	8	9	10	11	21
x^+	2	19	6	10	11	16	14	20	13	17	21
x^-	4	6	18	17	19	5	7	9	10	13	21

Table 3. Coordinates of rectangles in the rectangle visibility drawing of K_{22}

7 Conclusion

We have determined the multipartite number of several types of drawings. This significantly enlarges the class of graphs that are known to have such drawings. For example Fekete and Meijer show in [2] that each graph with at most 56 vertices has a 3-dimensional straight-line orthogonal box drawing. Comparing to this we prove that any 22-colorable graph has such a drawing.

Moreover the proofs of the existence of the drawing of the given type are constructive e.g. provide an exact description of a drawing of the specified multipartite graph. On the other hand the proofs provide only one possible drawing of the graph. It remains an open problem how to create a drawing that in addition fulfils some aesthetic criteria - minimizes the maximum length of an edge, the total length of the edges, total number of bends or the aspect ratio of the vertices (e.g. the ratio between the longest and the shortest side of the box that represents the vertex).

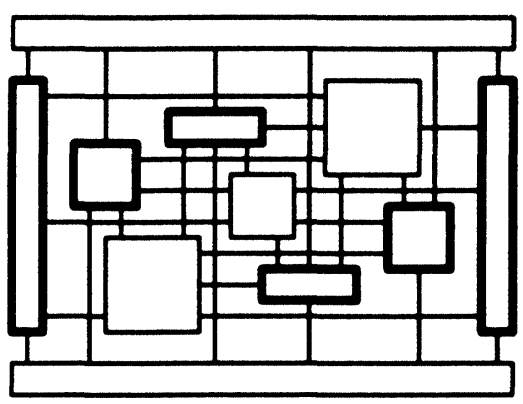


Figure 21: Straight-line 2D rectangle drawing of $K_{5,6}$

The proofs can be used to construct examples of multipartite graphs that don't have a drawing of the given type. Unfortunately (particularly due to the usage of Lemma 2.) the size of these examples is extremely big. We believe that the nice properties enforced by Lemmas 1., 2. and 3. are in fact necessary for a drawing of a graph with a large colorability and therefore the examples can be made reasonably big. For example, we have shown that the multipartite number of the straight-line 2D rectangle drawing is one, but how large are the bipartite graphs without such a drawing? Wood [1] shows that $K_{5,6}$ has (see Figure 21.) and that $K_{5,13}$ and $K_{6,9}$ don't have such a drawing. It is not known whether $K_{5,n}, 7 \leq n \leq 12$ and $K_{6,m}, 6 \leq m \leq 8$ admit a straight-line 2D rectangle drawing.

All graphs with the colorability lower or equal to the multipartite number have a drawing of the given type. Among the graphs with a larger colorability we can find some that don't have such a drawing, but it is an open problem

whether there exists $k \in \mathbb{N}$ such that no graph with the colorability equal to k has a 3D straight-line orthogonal box drawing. It seems that such k could be relatively small because there is no graph known that would have a colorability bigger than the largest known complete graph with the given orthogonal drawing. So, it is possible that the graphs that have an orthogonal drawing and have a large colorability also have a large complete graph as its subgraph.

The results presented in this work can be easily used to show that some graph has a drawing of the given type. It is sufficient to show that it can be colored by the specified number of colors – we don't need to determine the colorability of the graph, we can (for example) use a coloring produced by some heuristic algorithm. On the other hand it is not possible to use our results to show that a given graph doesn't have an orthogonal drawing. So, it would be desirable to find another large class of graphs and an algorithm that would decide whether the given member of this class has an orthogonal drawing or not.

References

[1] D. R. Wood (2000): Three-Dimensional Orthogonal Graph Drawing, *Ph.D. Thesis*, School of Computer Science and Software Engineering, Monash University

[2] S.P. Fekete, H. Meijer (1997): Rectangle and box visibility graphs in 3D, *Int. J. Comput. Geom. Appl.* **97**, 1-28

[3] S.P. Fekete, M.E. Houle, S. Whitesides (1995): New results on a visibility representation of graphs in 3D, *Proc. Graph Drawing* **95**, 234-241

[4] P. Bose, A. Dean, J. Hutchinson, T. Shermer (1996): On rectangle visibility graphs, *Proc. Graph Drawing* **96**, 25-44

[5] J.P. Hutchinson, T. Shermer, A. Vince (1999): On representations of some thickness-two graphs, *Comput. Geom.* **13(3)**, 161-171

[6] A.M. Dean, J.P. Hutchinson (1998): Rectangle-visibility Layouts of Unions and Products of Trees, *J. Graph Algorithms Appl.*, 1-21

[7] F.J. Cobos, J.C. Dana, F. Hurtado, A. Márquez, F. Mateos (1995): On a visibility representation of graphs, *Proc. Graph Drawing* **95**, 152-161

[8] P. Bose, H. Everett, S. Fekete, A. Lubiw, H. Meijer, K. Romanik. T. Shermer, S. Whitesides (1993): On a visibility representation for graphs in three dimensions, *Proc. Graph Drawing* **93**, 38-39

[9] R. Tamassia, I.G. Tollis (1986): A unified approach to visibility representations of planar graphs, *Discrete and Computational Geometry* **1**, 321-341

[10] S.K. Wismath (1985): Characterizing bar line-of-sight graphs, *Proc. ACM Symp. on Computational Geometry*, Baltimore. MD.

[11] T. Shermer (1996): Block visibility representations III: External visibility and complexity. In Kranakis, Fiala and Sack, eds., *Proc. of 8th Canadian Conference on Computational Geometry*, vol. 5 of *International Informatics Series*, Carleton University Press, 234-239

[12] Sándor P. Fekete, Michael E. Houle, Sue Whitesides (1997): The wobbly logic engine: proving hardness of non-rigid geometric graph representation problems, Report No. 97.273, Angewandte Mathematik und Informatik Universität zu Köln

[13] L. W. Beineke (1967): The decomposition of complete graphs into planar subgraphs. In F. Harary, ed., *Graph Theory and Theoretical Physics*, Academic Press, 139-154

Appendix

A Source Code

This appendix contains the source code of the Java implementation of the algorithm described in Lemma 9. The program starts with the four representations of K_2 shown on the Figure 13. (see lines 26-34) and attempts to enlarge the representation using the `next()` method. It examines all possible coordinates[2] of the sides of the next rectangle (lines 64-73) and checks if the boundary of the new rectangle intersects the boundaries of all previously added rectangles (76-94) and whether the rectangles can see each other through the line of visibility that corresponds to the intersection of the boundaries (95-118). Then it iterates through all possible orientations of the sides of the newly added rectangle (119-167) and tests (using the `goodOrientation()` method) if the new rectangle with the given orientation of sides can enlarge the current drawing of the complete graph.

```
 1: public class Colorability {
 2:   static int max; // Maximum size of found representation
 3:   static int[] N=new int[10]; // North coordinates
 4:   static int[] W=new int[10]; // West coordinates
 5:   static int[] E=new int[10]; // East coordinates
 6:   static int[] S=new int[10]; // South coordinates
 7:   static boolean[] Nup=new boolean[10]; // Orientation of north sides
 8:   static boolean[] Wup=new boolean[10]; // Orientation of west sides
 9:   static boolean[] Eup=new boolean[10]; // Orientation of east sides
10:   static boolean[] Sup=new boolean[10]; // Orientation of south sides
11:   // The following arrays contains only values array[i][j] for i>j
12:   static boolean[][] nes=new boolean[10][10]; // nes[i][j] determines whether i can see j
13:   static boolean[][] nws=new boolean[10][10]; // through the northeast corner of intersection
14:   static boolean[][] sws=new boolean[10][10]; // of i and j; nws, sws and ses have similar
15:   static boolean[][] ses=new boolean[10][10]; // meaning
16:   static boolean[][] ne1=new boolean[10][10]; // i-th east side intersects j-th north side?
17:   static boolean[][] ne2=new boolean[10][10]; // i-th north side intersects j-th east side?
18:   static boolean[][] nw1=new boolean[10][10]; // i-th north side intersects j-th west side?
19:   static boolean[][] nw2=new boolean[10][10]; // i-th west side intersects j-th north side?
20:   static boolean[][] sw1=new boolean[10][10]; // i-th west side intersects j-th south side?
21:   static boolean[][] sw2=new boolean[10][10]; // i-th south side intersects j-th west side?
22:   static boolean[][] se1=new boolean[10][10]; // i-th south side intersects j-th east side?
23:   static boolean[][] se2=new boolean[10][10]; // i-th east side intersects j-th south side?
24:
25:   public static void main(String args[]) {
26:     // All sides of the lowest rectangle are oriented up
27:     Nup[0]=Wup[0]=Sup[0]=Eup[0]=true;
28:     // Only one (w.l.o.g. west) side of the second lowest rectangle is oriented down
29:     Nup[1]=Sup[1]=Eup[1]=true; Wup[1]=false;
30:     // There are 4 possible positions of the second rectangle wrt the first one.
31:     N[0]=W[1]=S[0]=E[0]=0; N[1]=W[0]=S[1]=E[1]=1; next(2);
32:     N[0]=W[1]=S[1]=E[0]=0; N[1]=W[0]=S[0]=E[1]=1; next(2);
33:     N[0]=W[1]=S[0]=E[1]=0; N[1]=W[0]=S[1]=E[0]=1; next(2);
34:     N[0]=W[1]=S[1]=E[1]=0; N[1]=W[0]=S[0]=E[0]=1; next(2);
35:     System.out.println("MAX: "+max); // Report the maximum size
36:   }
37:
38:   // Checks whether we have a valid orientation of _where_ rectangle.
39:   // Expects where'th items of ne1, ne2, Nup etc. arrays to be filled.
40:   static boolean goodOrientation(int where) {
41:     for (int i=0; i<where; i++) {
42:       boolean ne = (ne1[where][i]&&Nup[i]&&!Eup[where])
43:                 || (ne2[where][i]&&!Nup[where]&&Eup[i]);
```

[2]The x^+, x^-, y^+ resp. y^- coordinate is called the east, the west, the north resp. the south coordinate in the source code.

```
44:       boolean nw = (nw1[where][i]&&Wup[i]&&!Nup[where])
45:                 || (nw2[where][i]&&!Wup[where]&&Nup[i]);
46:       boolean sw = (sw1[where][i]&&Sup[i]&&!Wup[where])
47:                 || (sw2[where][i]&&!Sup[where]&&Wup[i]);
48:       boolean se = (se1[where][i]&&Eup[i]&&!Sup[where])
49:                 || (se2[where][i]&&!Eup[where]&&Sup[i]);
50:       if (!((ne&&nes[where][i])||(nw&&nws[where][i])
51:           ||(sw&&sws[where][i])||(se&&ses[where][i]))) return false;
52:     }
53:     return true;
54:   }
55:
56:   // Adds another rectangle (if possible) into the existing
57:   // representation with _where_ rectangles.
58:   static void next(int where) {
59:     if (where == 9) {
60:         System.out.println("Representation of K_9 found!");
61:         System.exit(0);
62:     }
63:     if (where>max) max=where;
64:     for (int n=0; n<=where; n++) { // Find north coordinate
65:       N[where]=n;
66:       for (int i=0; i<where; i++) if (N[i]>=n) N[i]++;
67:       for (int w=0; w<=where; w++) { // Find west coordinate
68:         W[where]=w;
69:         for (int i=0; i<where; i++) if (W[i]>=w) W[i]++;
70:         for (int s=0; s<=where; s++) { // Find south coordinate
71:           S[where]=s;
72:           for (int i=0; i<where; i++) if (S[i]>=s) S[i]++;
73:           for (int e=0; e<=where; e++) { // Find east coordinate
74:             E[where]=e;
75:             for (int i=0; i<where; i++) if (E[i]>=e) E[i]++;
76:             boolean see = true;
77:             for (int i=0; i<where; i++) {
78:               // Checks that boundary of where intersects boundary of i
79:               ne1[where][i]=(E[i]>E[where])&&(N[i]<N[where]);
80:               ne2[where][i]=(E[i]<E[where])&&(N[i]>N[where]);
81:               nw1[where][i]=(N[i]>N[where])&&(W[i]<W[where]);
82:               nw2[where][i]=(N[i]<N[where])&&(W[i]>W[where]);
83:               sw1[where][i]=(W[i]>W[where])&&(S[i]<S[where]);
84:               sw2[where][i]=(W[i]<W[where])&&(S[i]>S[where]);
85:               se1[where][i]=(S[i]>S[where])&&(E[i]<E[where]);
86:               se2[where][i]=(S[i]<S[where])&&(E[i]>E[where]);
87:               see=see&&(ne1[where][i]||ne2[where][i]||nw1[where][i]||nw2[where][i]
88:                       ||sw1[where][i]||sw2[where][i]||se1[where][i]||se2[where][i]);
89:               if (!see) break; // The boundaries of where and i don't intersect
90:             }
91:             if (!see) { // The boundary of where doesn't intersect all existing boundaries
92:               for (int i=0; i<where; i++) if (E[i]>e) E[i]--;
93:               continue;
94:             }
95:             for (int i=0; i<where; i++) {
96:               nes[where][i]=nws[where][i]=sws[where][i]=ses[where][i]=true;
97:               for (int j=i+1; j<where; j++) {
98:                 // Determines nes, nws, sws and ses[where][*] values
99:                 boolean nn=(N[j]<N[i])&&(N[j]<N[where]);
100:                boolean ww=(W[j]<W[i])&&(W[j]<W[where]);
101:                boolean ee=(E[j]<E[i])&&(E[j]<E[where]);
102:                boolean ss=(S[j]<S[i])&&(S[j]<S[where]);
103:                nes[where][i]=nes[where][i]&&(nn||ee);
104:                nws[where][i]=nws[where][i]&&(nn||ww);
105:                sws[where][i]=sws[where][i]&&(ss||ww);
106:                ses[where][i]=ses[where][i]&&(ss||ee);
107:              }
108:              // Check whether _where_ can see i through intersecion of their boundaries
109:              see=see&&((nes[where][i]&&(ne1[where][i]||ne2[where][i]))
110:                      ||(nws[where][i]&&(nw1[where][i]||nw2[where][i]))
111:                      ||(sws[where][i]&&(sw1[where][i]||sw2[where][i]))
112:                      ||(ses[where][i]&&(se1[where][i]||se2[where][i])));
113:              if (!see) break;
114:            }
115:            if (!see) { // _where_ cannot see all rectangles through intersection of boundaries
116:              for (int i=0; i<where; i++) if (E[i]>e) E[i]--;
117:              continue;
118:            }
119:            // Find orientation of the added rectangle
120:            switch (where) {
121:              case 8: // Ninth => orient all sides down
122:                Nup[where]=Wup[where]=Sup[where]=Eup[where]=false;
123:                if (goodOrientation(where)) next(where+1); break;
124:              case 7: // Eighth => it is sufficient to orient at most one side up
125:                for (int i=0; i<5; i++) {
126:                  Nup[where]=Wup[where]=Sup[where]=Eup[where]=false;
127:                  switch (i) { // case 0: all sides oriented down
128:                    case 1: Nup[where]=true; break;
```

```
129:                     case 2: Wup[where]=true; break;
130:                     case 3: Sup[where]=true; break;
131:                     case 4: Eup[where]=true; break;
132:                   }
133:                   if (goodOrientation(where)) next(where+1);
134:                 }
135:                 break;
136:               default:
137:                 boolean[] ors = new boolean[16]; // Determines possible orientation of where
138:                 // 3rd to 6th rectangle => there are some rectangles below =>
139:                 // we cannot have all sides oriented up (e.g. i==0)
140:                 // we will try to add some rectangles above =>
141:                 // we cannot have all sides oriented down (e.g. i==15)
142:                 for (int i=1; i<15; i++) {
143:                   if (ors[i]) { // See comments in the if-branch below
144:                     if ((i&1)==0) ors[i+1] = true;
145:                     if ((i&2)==0) ors[i+2] = true;
146:                     if ((i&4)==0) ors[i+4] = true;
147:                     if ((i&8)==0) ors[i+8] = true;
148:                   } else {
149:                     // i to orientation conversion
150:                     Nup[where]=((i&1)==0);
151:                     Wup[where]=((i&2)==0);
152:                     Sup[where]=((i&4)==0);
153:                     Eup[where]=((i&8)==0);
154:                     if (goodOrientation(where)) {
155:                       // This orientation allows _where_ to see all rectangles below
156:                       next(where+1);
157:                       // We were able to see all rectangles below and therefore
158:                       // it doesn't have a sense to try orientations that arise
159:                       // from this one by making some up-oriented edges down-oriented
160:                       if (Nup[where]) ors[i+1] = true;
161:                       if (Wup[where]) ors[i+2] = true;
162:                       if (Sup[where]) ors[i+4] = true;
163:                       if (Eup[where]) ors[i+8] = true;
164:                     }
165:                   }
166:                 } // end of i-for statement
167:             } // end of switch statement
168:             for (int i=0; i<where; i++) if (E[i]>e) E[i]--;
169:           } // end of e-for statement
170:           for (int i=0; i<where; i++) if (S[i]>s) S[i]--;
171:         } // end of s-for statement
172:         for (int i=0; i<where; i++) if (W[i]>w) W[i]--;
173:       } // end of w-for statement
174:       for (int i=0; i<where; i++) if (N[i]>n) N[i]--;
175:     } // end of n-for statement
176:   }
177: }
```