Combining Constructive and Equational Geometric Constraint Solving Techniques

R. Joan-Arinyo and A. Soto-Riera
Universitat Politècnica de Catalunya
Departament de Llenguatges i Sistemes Informàtics
Av. Diagonal 647, 8a, E–08028 Barcelona

e-mail: [robert, tonis]@lsi.upc.es

Abstract

In the past few years, there has been a strong trend towards developing parametric, computer aided design systems based on geometric constraint solving. An efective way to capture the design intent in these systems is to define relationships between geometric and technological variables. In general, geometric constraint solving including functional relationships requires a general approach and appropiate techniques to achieve the expected functional capabilities.

This work reports on a hybrid method which combines two geometric constraint solving techniques: Constructive and equational. The hybrid solver has the capability of managing functional relationships between dimension variables and variables representing conditions external to the geometric problem. The hybrid solver is described as a rewriting system and is shown to be correct.

Keywords: Geometric constraint solving, constructive techniques, equational techniques, rewriting systems, canonical forms.

1 Introduction

In design and manufacturing applications, users of computer aided design systems are interested in defining functional relationships between dimension variables, since such relationships express design intent very flexibly. Some parametric relationships can be implemented by structuring the sketch appropriately, [8]. Moreover, simple functional relationships are the content of certain geometry theorems, such as the theorems of proportionality and many other classical results. Such theorems can be added to the constraint solver to extend its analysis

capabilities. But in general, geometric constraint solving including functional relationships between dimension variables requires a more general approach and requires appropriate techniques and tools to achieve those functional capabilities that users expect.

This work reports on a technique we have developed to combine constructive geometric constraint solvers with equational solvers in order to complement the advantages of each technique.

For surveys of the literature on geometric constraint solving see, e.g, [2, 8, 10, 12]. Briefly, the problem of integrating functional relationships and geometric constraints has been attempted by mapping both problems into a common representational domain. In particular, we can map both problems to a system of nonlinear equations and then solve the system. The system decomposition can be based on graph decomposition, [1].

Some authors analyze the equations using propagation techniques, [10, 12]. Broadly speaking, this mapping approach risks losing domain-specific information associated with the geometric constraint problem. Moreover, the decomposition and analysis techniques do assume that the resulting equations are independent, and this assumption may be violated on the geometry side where overconstrained problems may arise, [5, 16].

The approach we propose in this paper combines native constructive geometric constraint solving techniques with algebraic equation solving without mapping one problem domain to the other. Rather, we propose to combine a constructive, rule-based geometric constraint solver with a solver for functional relationships. Both solvers should remain largely independent of each other and proceed incrementally in parallel, with progress of each solver being posted to the other as the problem solution unfolds. This approach improves the work reported in [11] in two ways. First the hybrid solver is organised in a uniform way as a rewriting system. As a result, there is no need of an specific algorithm to control the information flow in the solver. Second, in the analysis of the functional relationships we introduce a new concept that we call *restriction* of a bigraph. This makes possible to subsume in just one rule all the rules dealing with equational analysis. Furthermore, it permits to extend the solver scope to consider certain class of problems involving geometric elements with more than two degrees of freedom, for instance, circular arcs and circles with unknown radii, without extending the repertoire of rules.

The contents of the paper is organised as follows. Section 2 provides basic definitions. Section 3 briefly describes the constructive solver. In Section 4 first we recall some fundamental concepts about bigraphs in connection with equations systems solving that we will make use of. Then we present our approach to equation analysis, and define the hybrid solver as a rewriting system. Sec-

tion 5 develops a simple case study to illustrate how our approach works. The correctness is proved in Section 6. Finally, Section 7 summarizes the work.

2 Symbolic Constraints

We will use the terminology defined in [11]. We consider geometric constraints from [5, 16]; that is, distance, angle, parallel, perpendicular, concentric, tangent, and so on. These constraints are extended by allowing symbolic constraints of distance and angle, where the "value" of the constraint is a variable symbol also called the *tag*.

A *valuated geometric constraint* is a distance or angle constraint whose value is a constant.

A *symbolic geometric constraint* is a distance or angle constraint whose value is a variable tag. When the value of the variable can be determined, the constraint is converted into a valuated constraint.

A *constraint equation* is an equation some of whose variables can be tags of symbolic constraints. We will refer to those variables involved in constraint equations which are not tags as *external variables*. In this paper we restrict to algebraic equations to simplify the theory of when a system of equations has a finite set of solutions, [20].

A *geometric constraint problem* consists of a finite set of geometric elements g_k, valuated and symbolic constraints between pairs of geometric elements, a set of variables, and a set F of constraint equations. We assume that if there is a subset of F which can be solved independently, it has ben resolved in a preprocessing step.

3 The Constructive Solver

The constructive solver considered here is a *rule-constructive* solver described in [13] and [16]. The solver handles bidimensional geometric configurations composed from points, segments, and arcs and circles with given radii. The constraints that can be defined on those objects include distance between two points, perpendicular distance between a point and a segment, and angle between two segments. Incidence, perpendicularity, parallelism, tangency and concentricity constraints can also be defined. Internally, these constraints are represented in terms of distances and angle constraints, [19]. The solver uses rewrite rules for the discovery of the construction steps and it is a *variational* solver, i.e., the solver processes the constraints without the need of arranging

them in a predefined ordering sequence. Furthermore, the solver can deal with geometric constraint problems with circular constraints.

3.1 Data Representation

All the constraints above mentioned can be represented by means of distance between two points, distance between a point and a straight segment and angle between two straigth segments. The notation used is derived from that reported by Verroust in [22]. The distance between points constraint is represented by means of a **CD** set, the point-segment distance constraint is represented by a **CH** set, and the angle between two segments is represented by a **CA** set. These sets are defined as follows.

A **CD** set is a set of points with mutually constrained distances. A frame of reference is attached to each **CD** set to which the points in the set are refered to. It is worth to note that a sketch is solved when all the points in the sketch belong to the same **CD** set. A **CH** set is a point and a segment constrained by the perpendicular distance from the point to the segment. A **CA** set is a pair of oriented segments which are mutually constrained in angle. We will refer generically to the **CD**, **CA** and **CH** sets as *constraint sets*.

3.2 Rules

Rules are classified depending on the functionality as creation rules, merging rules or construction rules.

Creation rules create **CD** sets, **CA** sets and **CH** sets by interpreting the geometric object defined by the user. The sign of the distances and angles are defined based on what the user has sketched. When a distance constraint between two points is given, a **CD** set is created. The position of the points in the associated frame of reference are $(0, 0)$ and $(d, 0)$. Whenever a point, a segment and the perpendicular distance from the point to the segment are given, a **CH** set is created.

Only one rule belongs to the merging rules type. The rule allows to compute the transitive closure of the angle constraint set. When a segment belongs to two different **CA** sets, ca_1 and ca_2, a new **CA** set, ca_3, is created which constrains in angle two segments, one from ca_1 and one from ca_2, both segments being different from the segment shared by ca_1 and ca_2.

Construction rules merge **CD** sets, **CH** sets, and **CA** sets into larger **CD** sets. Merging is performed by building triangles and a few quadrilaterals. A complete description of each rule can be found in [14].

3.3 Solver Architecture

The solver architecture follows a general architecture for constructive geometric constraint solving systems that has been proved to be useful when all the constraints defined by the user are valuated, [5, 11, 16]. This architecture splits the solution procedure into two main phases: The analysis phase and the construction phase.

In the analysis phase, first each single constraint defined by the user between two geometric elements is translated into a simple graph. Then, the analyzer performes a sequence of graph merging operations such that each operation corresponds to a specific geometric construction step. The problem is solvable if, at the end of the merging process, a single graph with all the geometric elements has been obtained. The output is a symbolic construction plan. We call this phase the *analyzer*.

In the construction phase the actual construction of the geometric elements is carried out by applying the construction plan determined by the analyzer to the parameters values assigned by the user. The construction is performed by solving certain standard sets of algebraic equations. This phase is known as the *constructor*.

3.4 The Solver as a Rewriting System

After Bruderlin, [6], and Dershowitz, [7], the idea behind solvers based on geometric rewrite rules is to replace some facts in the database by *simpler* ones. In the constructive solver discussed above, initially, the **CD** and **CH** constraint sets represent the sets of point-point and point-segment distance constraints derived from the constraint problem while the **CA** sets are the transitive closure of the angle constraints defined by the user. The solver starts by applying the constructive rules to these initial sets. Then the rules are repeatedly applied to the resulting constraint sets until either there is only one **CD** set which contains all the points in the sketch or no rule applies. In the first case the resulting **CD** set is a solution whereas in the second case the geometric constraint problem either does not define the geometric object consistently or is not solvable with the available set of rules. Every time a rule is triggered we will say that a reduction step has been performed.

In rewriting theory, a rule over a set of terms $\mathbf{T}$ is an ordered pair $< l, r >$ of terms, which are usually written as $l \rightarrow r$. It is said that l rewrites or reduces to r; [3, 7, 21]. All the construction rules in the constructive solver considered can be expressed as rewriting rules where the terms on the left side are constraint sets and the term on the right side is always a **CD** set. That is, the construction rules $\mathbf{CR}_l \rightarrow \mathbf{CR}_r$, are such that $\mathbf{CR}_l = \{\mathbf{CX}_1, \mathbf{CX}_2, \mathbf{CX}_3\}$,

where $\mathbf{CX}_i$ is either a **CD** set or a **CH** set or a **CA** set, and $\mathbf{CR}_r$ is a **CD** set. The set of rewriting rules, denoted by $\rightarrow_\rho$, can be found in [14].

Denoting by $\mathbf{C}_i$ the term whose members are the constraint sets after having applied the i-th reduction step, the analyzer can be represented by the pair $(\mathbf{C}_i, \rightarrow_\rho)$ which is a *reduction system*, [3, 7, 21]. Assuming that the geometric constraint problem does define the geometric object consistently, the correctness of the analyzer is established in [15]. That is, 1) The analyzer terminates after a finite number of reduction steps, and 2) The sequence in which the rules are applied does not matter for the result.

4 The Hybrid Solver

The hybrid solver is built by extending both the data representation and the set of rules available in the constructive solver of Section 3. Data representation is extended to accomodate equational constraints. A new rule is provided to deal with them. Besides that, the set of geometric elements considered in the basic constructive solver is extended with circular arcs and circles with unknown radii. Following [19], incidence, tangency and concentricity constraints defined on these geometric elements are translated into distances and angle symbolic constraints involving the centers and radii.

Before describing the main elements in the hybrid solver we recall a mathematical tool that will play a central role in our approach.

4.1 Bigraphs and Systems of Equations

As a basic technique for reasoning about systems of equations we will use bigraphs. Here we recall the most relevant aspects of bigraphs. For an in-depth study see [18], and for bigraphs in equations systems solving see [20].

Let F be the set of equations, X the set of all variables occurring in the equations, and let E be the set of edges defined by the pairs (f, x), with $f \in F$ and $x \in X$ such that the variable x occurs in the equation f. Then $B = (E, F, X)$ is the *bigraph* associated with the set of constraint equations.

Let $B = (E, F, X)$ be a bipartite graph. The vertex set F is called the *entrance* and the vertex set X the *exit*. A *Menger-type linking* from F to X is defined as a set of pairwise vertex-disjoint directed paths from a vertex in F to a vertex in X. The size of a linking is defined to be the number of directed paths from F to X contained in the linking. A linking of the maximum size is called a *maximum linking* and, if $|F| = |X|$, a linking of size $|F|$ is called a *complete linking*. With these concepts, the bipartite graph B has a unique decomposition into a set of

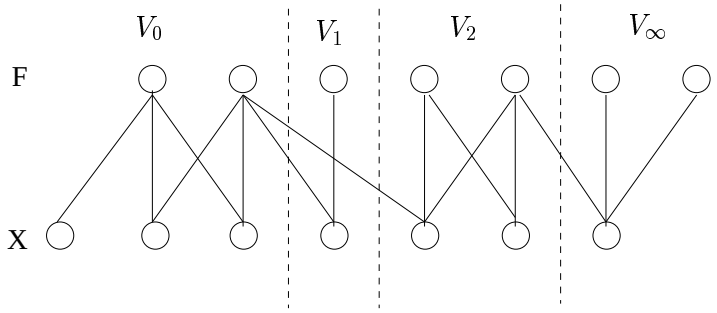

Figure 1: M-components of a bigraph $B = (E, F, X)$.

induced subgraphs, namely, $\{V_0, V_1, \ldots, V_r, V_\infty\}$, with a partial order $\prec$ defined on it, [20]. The decomposition along with the partial order $\prec$ is called the *M-decomposition* of $B = (E, F, X)$ with respect (F, X), [20]. Subgraphs V_i are not necessarily disjoint with respect each other. Each V_i is called an *M-irreducible component* or just an *M-component*, $\{V_i | i = 1, \ldots, r\}$ is the consistent part, V_0 is the *minimal inconsistent part*, and V_∞ is the *maximal inconsistent part*. Figure 1 illustrates these concpets, [20].

The M-decomposition of the bigraph $B = (E, F, X)$ associated with the equations F, permits to study the solvability of F. The set of equations F is structurally solvable if, and only if, both V_0 and V_∞ are the empty set. An M-component $V_i, 1 \leq i \leq r$, in the consistent part corresponds to a subset of equations in F which is structurally solvable and cannot be decomposed further with the structural solvability maintained. It has a structure that admits a unique solution if the values of all the variables belonging to V_j such that $V_j \prec V_i$ are determined. The subsets of equations corresponding to the inconsistent parts, V_0 and V_∞, if they exist, are not solvable. The problem corresponding to V_0 is underdetermined, i.e., has more unknowns than equations, and that to V_∞ is overdetermined, i.e., has fewer unknowns than equations.

4.2 Data Representation

In the hybrid solver, we need to represent three different types of data: Valuated geometric constraints, symbolic geometric constraints, and equations.

Valuated geometric constraints are represented in the same way as in the constructive solver; i.e., by the constraint sets, **CD**, **CA** and **CH** sets.

Symbolic geometric constraints are represented by constraint sets **CD**, **CA** and **CH**, where the constraint value is a tag. Symbolic geometric constraints are also translated into an equational representation. The variables in each equation are

the constraint tag and the coordinates of the points involved in the geometric constraint, in what follows refered to as *geometric variables*.

Equations are represented by a bigraph defined as follows. Let F_g be the set of equations generated by the symbolic geometric constraints, and let X_g be the set of geometric variables and tags in F_g. Let F_c be the set of constraint equations in the geometric constraint problem and let X_c be the set of variables occurring in F_c. Note that tags can occur in both X_c and X_g. Let $F = F_c \cup F_g$ and $X = X_c \cup X_g$, and let E be the set of edges defined by the pairs (f, x), with $f \in F$ and $x \in X$ such that the variable x occurs in the equation f. Then $B = (E, F, X)$ is the *bigraph* associated with the initial set of equational relationships.

4.3 Equations Analysis

Equation analysis is performed using the M-decomposition of the bigraph B associated with the set of equations F. When applied to systems of equations, a convenient partial order between M-components is defined by $V_\infty \prec V_i, 1 \leq i \leq r, \prec V_0$, [20].

As pointed out in Section 2, we assume that the initial set of equations F does not contain a subset of equations which can be solved independently. Therefore, at least, one variable which is a tag in a symbolic geometric constraint, along with the corresponding geometric variables, will always occur in every possible subsytem of equations.

Since geometric variables represent coordinates (degrees of freedom) of geometric elements, they only can be evaluated with respect to a frame of reference. Thus, we seek some **CD** set C in the constraint problem such that will allow to valuate some geometric variables with respect to its local frame of reference.

We define now the concept on which the algorithm that effectively performs the equational analysis is based.

Let B be the bigraph associated with a geometric constraint problem and let C be a **CD** set. We define the bigraph $R(B, C)$ as the subgraph of B resulting from valuating with respect the local frame of reference of the **CD** set C, those geometric variables in B generated by symbolic geometric constraints such that the geometric elements on which they are defined belong to C. The bigraph $R(B, C)$ is called the *restriction* of bigraph B by the **CD** set C. Figure 2 illustrates this definition. Figure 2a shows a simple geometric constraint problem with two symbolic constraints and one constraint equation. Symbolic constraints x and y define respectively the distance between points p_1 and p_3, and between points p_3 and p_4. The constraint equation is $x - 2y = 1.5v$ where v is an external variable. Figure 2b shows the associated bigraph. Figure 2c shows the

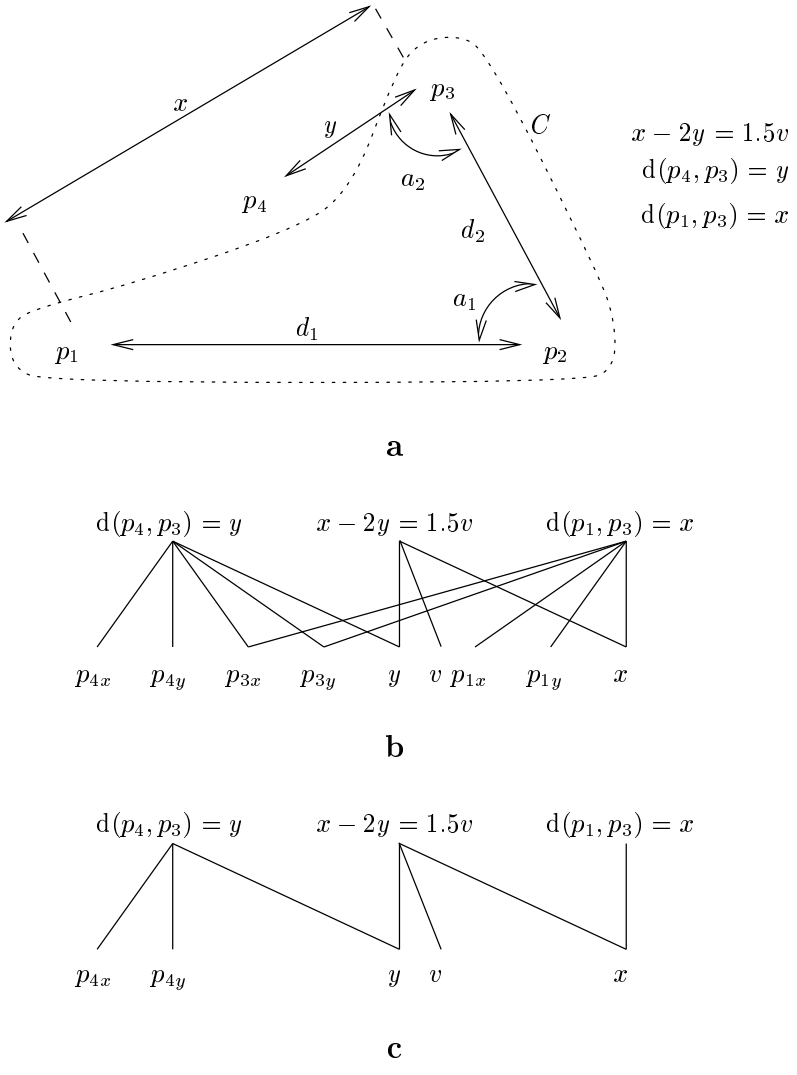

Figure 2: Restriction of a bigraph by a **CD** set C.

restriction of the bigraph by the **CD** set C in the constraint problem.

Note that since geometric variables do not occur in constraint equations, constraint equations do not result affected by restrictions. Furthermore, since in each equation generated by a symbolic constraint there is a tag involved which is not valuated by any restriction of B, restrictions do not eliminate geometric equations from the bigraph.

In these conditions, we try to identify a subset of equations that is solvable by applying the M-decomposition to $R(B, C)$. Specifically, the computational

steps carried out for a given **CD** set C in the constraint problem are:

1. Compute $R(B, C)$, the restriction of B by the **CD** set C.
2. Identify a solvable subsystem S from the M-decomposition of bigraph $R(B, C)$.
3. Solve the identified subsystem.
4. For each computed variable x that is tag of a symbolic constraint whose equation is not in S, valuate the corresponding symbolic constraint.
5. For each pair of computed variables (p_x, p_y) that fix a point p in the local frame of reference of the **CD** set C, add the point p to the set C.
6. Valuate in the bigraph each computed external variable.
7. Compute the new bigraph as the subgraph of B induced by the set of equations $F - S$.

We will refer to this sequence of symbolic computations as a *computable* step, [11].

4.4 Characterization of the Scope Extension

Characterizing the solver scope extension amounts to identify those situations where the degrees of freedom of the new geometric elements can be cancelled. For the shake of conciseness let us consider just circular arcs and circles with unknown radii as the new geometric elements. Note that they have three degrees of freedom.

In our representation, geometric constraints placed on each circular arc and circle with unknown radius are translated into symbolic constraints which depend on the center point and the radius. Hence, cancelling the three degrees of freedom means to determine the center and radius of the geometric element.

For analysis purposes, each symbolic constraint is also represented as an equational relationship depending on center points and radii. In the case of circles and circular arcs with unknown radii, a center point is represented by two variables corresponding to its two components, and the radius by one variable.

Since geometric constraints with symbolic values cannot be valuated by constructive steps, valuating the center point and radius necessarily involves the resolution of a subsystem of equations of the geometric constraint problem. Therefore, the new geometric constraint problems that are solvable are those such that for each circular arc and circle with unknown radius there exists an

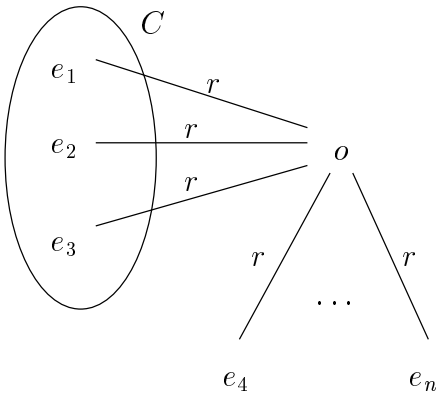

Figure 3: Constraint sets.

analysis step where the equational analysis splits de associated bigraph into an empty minimal inconsistent part plus an M-component that valuates the coordinates of the center with respect some **CD** set, and valuates the radius.

Figure 3 shows a set of constraints placed between a circle with unknown center o and unknown radius r, and the geometric elements $e_1, e_2, \ldots, e_n$. Assume that there is a constructive step that generates a constraint set C which includes only three symbolic geometric constraints, say e_1, e_2, e_3. Assume that B is the bigraph associated with the subproblem involving the circle and geometric elements $e_1, e_2, \ldots, e_n$. Morover, let $R(B, C)$ be the restriction of bigraph B by the **CD** set C. Then, the M-decomposition of $R(B, C)$ is as illustrated in Figure 4. The consistent part labeled V_1 valuates both o and r. Figure 5 shows the resulting constraint sets.

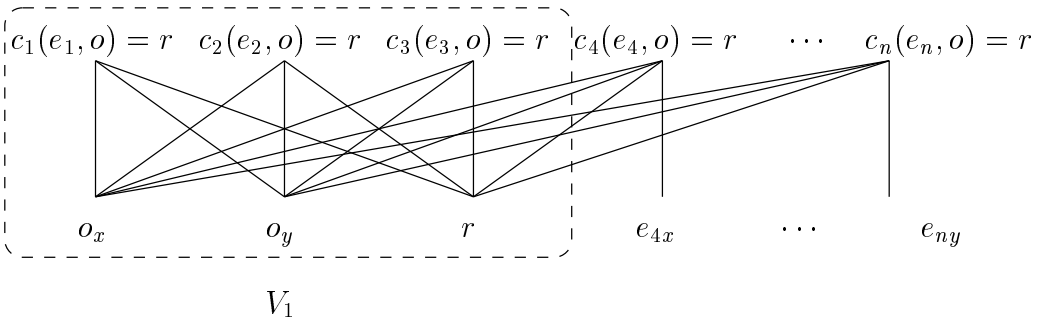

Figure 4: M-decomposition of restriction $R(B, C)$.

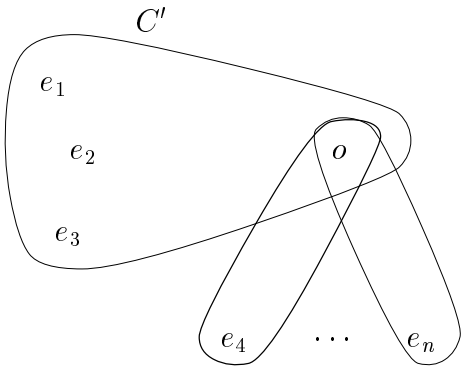

Figure 5: Constraint sets after fixing circle center and radius.

4.5 The Rules

Consistently with the data representation extension, the set of rules in the hybrid solver is the set of rules available in the constructive solver plus a rule which performes the analysis of the equations either in the bigraph B or in some restriction $R(B, C)$ of B.

We call this rule a *computational* rule, [11], and we denote it by $\rightarrow_\kappa$. Since the set of rules in the constructive solver is $\rightarrow_\rho$, the set of available rules now is $\{\rightarrow_\rho, \rightarrow_\kappa\}$.

4.6 The Hybrid Solver as a Rewriting System

Let $\mathbf{C}_0$ be the set of initial constraint sets in the geometric constraint problem, and let B_0 be the bigraph associated with the initial set of equations.

The hybrid solver starts by applying the rules to the initial data representation $\mathbf{T}_0 = (\mathbf{C}_0, B_0)$. Then the rules are repeatedly applied to the resulting data representation until either there is only one **CD** set which contains all the points in the geometric constraint problem or no rule applies. From a functional point of view, no priority is assigned to any rule.

Denoting by $\mathbf{T}_i = (\mathbf{C}_i, B_i)$ the data representation after having applied the i-th reduction step, the hybrid analyzer can be represented by $(\mathbf{T}_i, \rightarrow_\rho, \rightarrow_\kappa)$ which is a reduction system, [3, 7, 21].

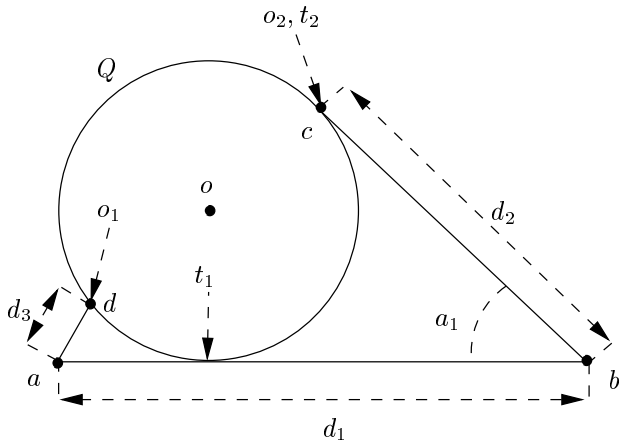

Figure 6: A geometric constraint problem.

5 Case Study

We illustrate how the solver works by showing a complete, simple case study. Figure 6 shows the geometric constraint problem to be solved. The set of geometric elements contains the points a, b, c and d; the segments (a, b), (a, d) and (b, c); and the circle Q with center o. The valuated constraints are the distances $\mathrm{d}(a, b) = d_1$, $\mathrm{d}(b, c) = d_2$ and $\mathrm{d}(a, d) = d_3$, and the angle $\mathrm{a}((a, b), (b, c)) = a_1$. The circle Q has an unknown radius and the geometric constraints placed on it are: Segments (a, b) and (b, c) will be tangent to the circle, $t_1 \equiv \mathrm{t}(Q, (a, b))$ and $t_2 \equiv \mathrm{t}(Q, (b, c))$, and points d and c will be coincident with the circle, $o_1 \equiv \mathrm{on}(d, Q)$, $o_2 \equiv \mathrm{on}(c, Q)$. Tangency and coincidence constraints are translated into distance and angle constraints. Since the radius of the circle is unknown, the constraints are translated into symbolic constraints involving the center, o, and radius, r, of the circle Q, as proposed in [19]. Tangency conditions $t_1 \equiv \mathrm{t}(Q, (a, b))$ and $t_2 \equiv \mathrm{t}(Q, (b, c))$ are translated into symbolic point-segment perpendicular distance constraints $\mathrm{h}(o, (b, c)) = r$ and $\mathrm{h}(o, (a, b)) = r$. Coincidences are translated into the symbolic distance constraints $\mathrm{d}(o, c) = r$ and $\mathrm{d}(o, d) = r$.

Initially, a **CD** set and a **CA** set is created for each valuated distance constraint and each valuated angle constraint. The initial set of constraint sets is $\mathbf{C}_0 = \{CD_1, CD_2, CD_3, CA_1\}$. It is shown in Figure 7. **CD** sets are depicted by a dashed line enclosing the points contained in it, and a **CA** is represented by a dotted arc between a pair of segments.

The occurrence of variables and tags in equations and symbolic constraints is represented by a bigraph. The initial bigraph B_0 is shown in Figure 8. Note

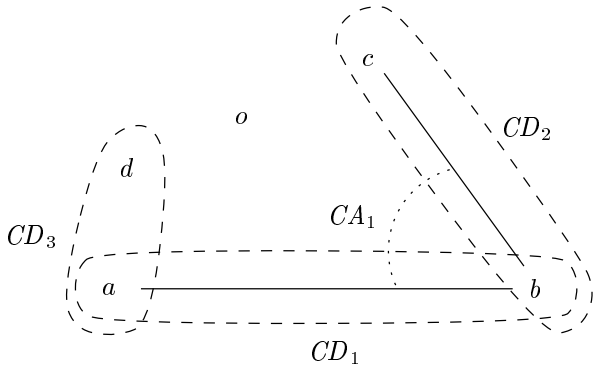

Figure 7: Initial set of constraint sets $\mathbf{C}_0$.

that a pair of geometric variables has been created for each geometric element occurring in a symbolic constraint.

The initial state of the solver can be represent by the term $\mathbf{T}_0 = (\mathbf{C}_0, B_0)$. The only rule that applies is a ρ rule that merges the constraint sets CD_1, CD_2 and CA_1 in, say, CD_4. See Figure 9. Since ρ rules do not affect the bigraph, the state of the solver, is now represented by the term $\mathbf{T}_1 = (\mathbf{C}_1, B_0)$ whith $\mathbf{C}_1 = \{CD_3, CD_4\}$.

Now no ρ rule applies to term $\mathbf{T}_1$. But the κ rule can be applied because the M-decomposition of the restriction of bigraph B_0 by constraint set CD_4, $R(B_0, CD_4)$, generates a non empty consistent part. Figure 10 shows the restriction $R(B_0, CD_4)$ and the M-components enclosed in dashed lines. The consistent component is V_1 and the minimal inconsistent component is V_0. Since the maximal inconsistent part is empty, the consistent part can be solved for the geometric variables o_x, o_y, and for the tag r, fixing the center and radius of circle Q. Note that the center is fixed with respect the **CD** set CD_4.

In general, a computable step has an effect on both the set of constraint sets and the bigraph. In this case, since the tag of a symbolic constraint, r, has been valuated, a new **CD** set, CD_5, has been created. Moreover, since geometric

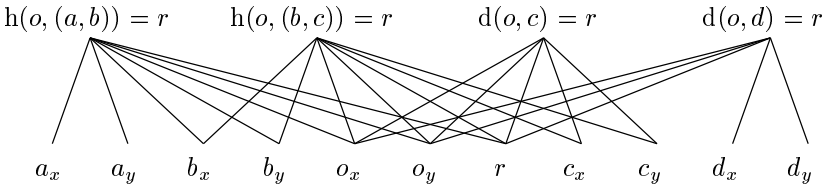

Figure 8: Initial bigraph B_0.

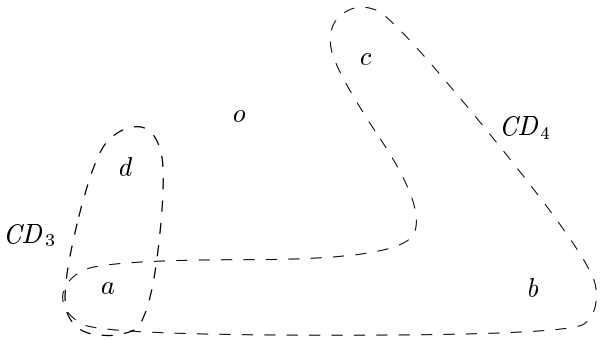

Figure 9: Set of constraint sets $\mathbf{C}_1$ after applying a ρ rule.

variables (o_x, o_y) fixing the point o with respect the **CD** set CD_4 has been valuated, this point o has been included in the set CD_4 producing CD_4'. The resulting set of constraint sets, shown in Figure 11, is $\mathbf{C}_2 = \{CD_3, CD_4', CD_5\}$. The new bigraph B_1 is the empty graph because once the solved equations and valuated tags are deleted from the bigraph B_0, results an empty set. The state of the solver is now represented by the term $\mathbf{T}_2 = (\mathbf{C}_2, B_1)$.

Finally, a ρ rule that merges CD_3, CD_4' and CD_5 into CD_6 can be applied. The result is the term $\mathbf{T}_3 = (\mathbf{C}_3, B_1)$. Since $\mathbf{C}_3$) is a singleton, CD_6, that contains all the geometric elements, and the bigraph B_1 is empty, the geometric constraint problem has been successfully solved.

6 Correctness

Following [11], we show that the algorithm is correct in the following sense. Let ρ denote a geometric reduction step, and κ the evaluation of a computable step.

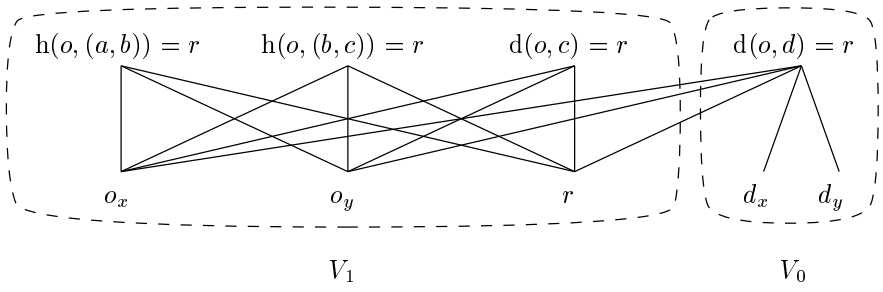

Figure 10: Restriction of B_0 by CD_4 and M-decomposition (V_1, V_0).

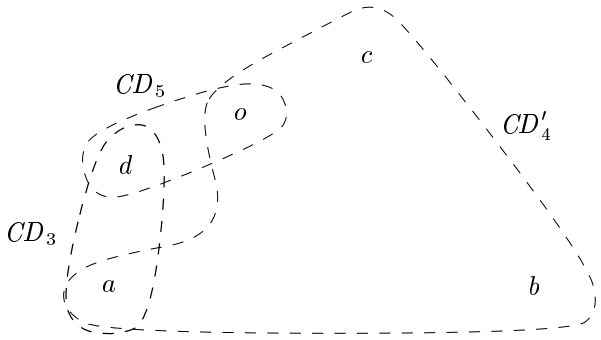

Figure 11: Set of constraint sets $\mathbf{C}_2$ after applying the κ rule.

We will prove the following statement:

> If there exists a sequence of steps of type ρ and κ that reduces the initial constraint sets to a single **CD** set and the bigraph to the empty graph, then the algorithm finds such a sequence.

Note that, a priori, there could be many different sequences, and that some of them could result in unsuccessful termination of the algorithm. To argue that this cannot happen, we need to introduce some definitions.

6.1 Definitions

In the correctness proof of the algorithm, we will use the terminology of [9], [11], and [15]. We recall this terminology here for the sake of completeness.

The *geometric constraint graph* associated with a geometric constraint problem is a graph $G = (C, V)$, where V is the set of geometric elements of the problem, and C is the set of geometric constraints. Associated with the graph is the set $\mathbf{C} = \{Z | Z \subset V\}$ of constraint sets. The constraint sets are a set cover of V but not a disjoint set cover.

Intuitively, a constraint problem can be overconstrained in one of three ways: First, discounting the symbolic constraints and the constraint equations, the constraint graph could be structurally overconstrained. Second, ignoring the geometric constraints, the system of equations supplied could contain an overdetermined subsystem. Third, the interaction of valuated constraints, symbolic constraints, and constraint equations introduces additional constraints, by valuating symbolic constraints, such that at some time an overconstrained, partially solved problem is obtained. Definitions below only consider the first and the third possibility. Accounting thereafter for the second possibility is easy.

In [15], a geometric constraint graph is structurally overconstrained if there is a vertex-induced subgraph with m vertices and more than $2m-3$ edges, [17]. In particular, the constraint graph itself cannot be well-constrained if it has more than $2n-3$ edges, where n is the number of vertices. In order to be usefull in our context, these definitions are generalized as follows, [11].

Definition 6.1 A geometric constraint problem is geometrically overconstrained if, for some derivable term $\mathbf{T}$, the geometric constraint graph $G=(C,V)$ associated with some constraint set in $\mathbf{T}$ is structurally overconstrained.

Definition 6.2
A geometric constraint problem is geometrically underconstrained if it is not geometrically overconstrained and for some derivable term $\mathbf{T}$, the geometric constraint graph $G=(C,V)$ associated with some constraint set in $\mathbf{T}$ is structurally underconstrained.

Definition 6.3
A geometric constraint problem is geometrically well-constrained if for every derivable term $\mathbf{T}$, the geometric constraint graph $G=(C,V)$ associated with every constraint set in $\mathbf{T}$ is structurally well-constrained.

In what follows, given a term $\mathbf{T}=(\mathbf{C},B)$, we will refer to the constraint graph associated with the set of constraint sets $\mathbf{C}$ as the constraint graph associated with $\mathbf{T}$.

6.2 Valuating by Computation

When valuation by computation is performed using the bigraph, only tags of symbolic constraints and external variables are valuated. Consistently, those symbolic geometric constraints whose tag result evaluated are updated as valuated constraints and added to the geometric constraint graph. The valuation of external variables does not affect the constraint sets. However, valuating external variables will have an effect on the bigraph because valuated variables are no longer unknown and, consequently, must be removed from the bigraph.

When valuation by computation is peformed with respect to a **CD** set C by solving a subset of equations in $R(B,C)$, besides symbolic constraints tags and external variables, geometric variables are valuated too. Therefore, those geometric elements that result fixed with respect to the **CD** set C, are included in C.

The valuation by computation process is called a κ-reduction.

6.3 Uniform Termination Property

Here we shown that the reduction system $(\mathbf{T}_0, \rightarrow_\rho, \rightarrow_\kappa)$ is terminating; that is, the analyzer terminates after a finite number of reduction steps.

Proposition 6.1
The reduction system $(\mathbf{T}_0, \rightarrow_\rho, \rightarrow_\kappa)$ is terminating.

Proof Assume that the initial term $\mathbf{T_0}$ has associated a constraint graph with n nodes, where n_v edges correspond to valuated constraints and n_s edges correspond to symbolic constraints.

The κ reduction does not add new nodes to the constraint graph. Let n_e be the number of external variables in the set of equations. Since each κ reduction step reduces the number of variables with uknown value by at least one, the total number of such reductions is bounded by $n_s + n_e$. Each ρ reduction step reduces the number of constraint sets by 2. Each κ-reduction, moreover, adds as many constraint sets to the constraint graph as there are symbolic constraints tags solved by the reduction. Thus, every reduction sequence has length less than $(n_v + n_s + 1)/2 + n_s + n_e$. □

6.4 Unique Normal Form Property

Having established termination, we begin by showing that if the problem is not geometrically overconstrained, then ρ-reductions always commute with κ-reductions.

Let $\mathbf{T}_0 = (\mathbf{C}_0, B_0)$ be the initial term. By [15], the rewrite system $(\mathbf{C}_0, \rightarrow_\rho)$ is terminating and Church-Rosser if the geometric constraint graph associated with $\mathbf{C}_0$ is not structurally overconstrained. In particular, two ρ reductions commute. Furthermore, if $\mathbf{C}_1$ and $\mathbf{C}_2$ are terms such that $\mathbf{C}_1 \rightarrow_\rho \mathbf{C}_2$ and $\mathbf{C}_1$ has a well-constrained associated geometric constraint graph, then the geometric constraint graph associated with $\mathbf{C}_2$ is also well-constrained.

Lemma 6.2
If $\mathbf{T_1} \rightarrow_\rho \mathbf{T_2}$ and $\mathbf{T_1} \rightarrow_\kappa \mathbf{T_2'}$, then there is $\mathbf{T_3}$ such that $\mathbf{T_2} \rightarrow_\kappa \mathbf{T_3}$ and $\mathbf{T_2'} \rightarrow_\rho \mathbf{T_3}$.

Proof See Figure 12. Let $R(B_1, C_1)$ be the restriction where the computationl step κ is carried out. The valuation by κ-reduction adds a number of constraint sets, the corresponding valuated constraints, to the associated geometric constraint graph, and a possibly empty set of points to the **CD** set C_1. Conversely, a ρ-reduction does not change the bigraph or the geometric constraint graph.

Assume that the κ-reduction only valuates tags and external variables. Using

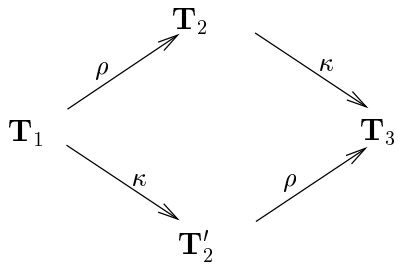

Figure 12: ρ-reductions and κ-reductions commute.

the same reductions, we have in general $\mathbf{T'_2} \rightarrow_\rho \mathbf{T_3}$ and $\mathbf{T_2} \rightarrow_\kappa \mathbf{T'_3}$. Let B_1, G_1 be the bigraph and constraint graph associated with $\mathbf{T}_1$. Analogously, B'_2, G'_2 are associated with $\mathbf{T}'_2$, and B'_3, G'_3 are associated with $\mathbf{T}'_3$. Clearly, B_1, G_1 are also associated with $\mathbf{T}_2$, and B'_2, G'_2 are also associated with $\mathbf{T}_3$. Let S_κ be the subset of equations solved in $R(B_1, C_1)$, and B_κ be the subgraph of B_1 induced by the set of equations $F_1 - S_\kappa$. Furthermore, let G_κ be the constraint graph associated with the symbolic constraints whose tags have been valuated in the κ-reduction. Then $B'_2 = B'_3 = B_\kappa$ and $G'_2 = G'_3 = G_1 \cup G_\kappa$. That is, states $\mathbf{T}_3$ and $\mathbf{T}'_3$ clearly have the same constraint graph and the same bigraph. Therefore $\mathbf{T}_3 = \mathbf{T}'_3$.

Finally assume that the κ-reduction also valuates geometric variables. If the ρ-reduction does not make use of the **CD** set C_1, the rational above trivially applies. Otherwise just note that the κ-reduction does not remove geometric elements in C_1, therefore the ρ-reduction still applies. □

Corollary 6.3
Let $\mathbf{T_1} \rightarrow_\rho \mathbf{T_2} \rightarrow_\kappa \mathbf{T_3}$ and $\mathbf{T_1} \rightarrow_\kappa \mathbf{T'_2}$. Then $\mathbf{T'_2}$ is overconstrained if, and only if, $\mathbf{T_3}$ is.

Proof The statement is trivial if $\mathbf{T_1}$ has a structurally overconstrained constraint graph. By [15], if the constraint graph of $\mathbf{T_1}$ is not structurally overconstrained, then neither is the constraint graph of $\mathbf{T_2}$.

Assume that the constraint graph of $\mathbf{T'_2}$ is structurally overconstrained. Then there exists a nonempty set of equations $\{f_1, \ldots, f_m\}$ defining $m > 0$ valuated constraints which have been added by reduction κ to the constraint graph G_1 of $\mathbf{T_1}$. By Lemma 6.2, ρ and κ commute. Hence, $\{f_1, \ldots, f_m\}$ defines the same $m > 0$ valuated constraints in $\mathbf{T_2}$, yielding a structurally overconstrained constraint graph G_3 of $\mathbf{T_3}$.

Now assume that the graph of $\mathbf{T_3}$ is structurally overconstrained. This implies that reduction κ adds at least one valuated constraint to the constraint graph G_2

of $\mathbf{T_2}$ to give a structurally overconstrained graph G_3. Since ρ and κ commute, the same set of valuated constraints can be added by reduction κ to the well-constrained graph G_1, resulting in an overconstrained graph G_2' associated with $\mathbf{T_2'}$. □

Lemma 6.4
Assume that $\mathbf{T_1}$ is not geometrically overconstrained. If $\mathbf{T_1} \rightarrow_{\kappa 1} \mathbf{T_2}$ and $\mathbf{T_1} \rightarrow_{\kappa 2} \mathbf{T_2'}$, then there is a $\mathbf{T_3}$ such that $\mathbf{T_2'} \rightarrow_{\kappa 1} \mathbf{T_3}$ and $\mathbf{T_2} \rightarrow_{\kappa 2} \mathbf{T_3}$.

Proof Let B_1 be the bigraph associated with $\mathbf{T_1}$. If both κ_1 and κ_2 are applicable to $\mathbf{T_1}$, then two different restrictions $R(B_1, C_1)$ and $R(B_1, C_2)$ can be defined in the bigraph B_1. Let S_{κ_1} and S_{κ_2} be, respectively, the subset of equations solvable in each restriction.

First assume that the sets of variables occurring in S_{κ_1} and in S_{κ_2} have an empty intersection. Then there are two independent solvable sets of equations each corresponding to a different restriction. Clearly the reductions commute.

For a contradiction now assume that S_{κ_1} and S_{κ_2} have variables in common. Since tags in geometric equations are coupled with external variables through constraint equations, at least one geometric symbolic constraint is shared by S_{κ_1} and S_{κ_2}. Let us denote it by x. Since S_{κ_1} and S_{κ_2} are two different systems of equations, the value computed for x in each system is, in general, different. Therefore the geometric constraint problem would be overconstrained. □

Theorem 6.5
For problems that are not geometrically overconstrained, the rewrite system $(\mathbf{T_0}, \rightarrow_\rho, \rightarrow_\kappa)$ is terminating and has the Church-Rosser property, that is the system generates unique normal forms.

Proof Direct consequence of the lemmas above. □

7 Summary

We have presented a hybrid technique for solving geometric constraint problems with the capabiliy of managing problems involving functional relationships between dimensions and external variables. The method combines constructive and equational approaches and is especially effective for constructive geometric constraint solvers that use the rewriting rule paradigm.

Our technique works on two sets of data, the geometric constraint data, and the symbolic equation data. Geometric data are represented by a set of constraint

sets. Symbolic equations are represented by a bigraph. The information flow between these two structures is managed by a new rewriting rule: the κ-reduction rule. This rule valuates symbolic constraints and external variables by solving a subset of constraint equations from the bigraph and adds the resulting valuated constraints to the set of constraint sets.

It has been shown that when the constraint problem is not overconstrained, the method is correct. That is, if there is a sequence of rewriting steps that reduces the constraint graph to a single **CD** set and the bigraph to the empty graph, then the method finds such a sequence. Since the rules can be applied in any order, strategies to conveniently optimize the analysis process can be devised.

Acknowledgement

The authors have been supported in part by the CICYT Spanish Research Agency under Grant TIC95-0630-C05-04.

References

[1] S. Ait-Auodia, R. Jegou, and D. Michelucci. Reduction of constraint systems. In *Compugraphic*, pages 83–92, Alvor, Portugal, 1993.

[2] R. Anderl and R. Mendgen. Parametric design and its impact on solid modeling applications. In *ACM Solid Modeling'95*, pages 1–12, Salt Lake City, Utah USA, 1995.

[3] L. Bachmair. Proof methods for equational theories. Technical report, Department of Computer Science, SUNY at Stony Brook, Stony Brook, New York 11794, 1991.

[4] W. Bouma, I. Fudos, C. Hoffmann, J. Cai, and R. Paige. A geometric constraint solver. Technical Report CSD-TR-93-054, Department of Computer Scineces, Purdue University, 1993.

[5] W. Bouma, I. Fudos, C. Hoffmann, J. Cai, and R. Paige. Geometric constraint solver. *Computer Aided Design*, 27(6):487–501, June 1995.

[6] B.D. Brüderlin. *Rule-Based Geometric Modelling*. PhD thesis, Institut für Informatik der ETH Zürich, 1988.

[7] N. Dershowitz and J.-P. Jouannaud. *Handbook of Theoretical Computer Science*, chapter Rewrite Systems, pages 244–320. Elsevier Science Publishers B.V., 1990.

[8] I. Fudos. Editable representations for 2D geometric design. Master's thesis, Purdue University, Department of Computer Sciences, 1993.

[9] I. Fudos and C.M. Hoffmann. Correctness proof of a geometric constraint solver. *International Journal of Computational Geometry and Applications*, 6(4):405–420, 1996.

[10] D. Henderson. *Constraint Management for Collaborative Design.* PhD thesis, Purdue University, Department of Mechanical Engineering, 1996.

[11] C.M. Hoffmann and R. Joan-Arinyo. Symbolic constraints in constructive geometric constraint solving. *Journal of Symbolic Computation*, 23:287–300, 1997.

[12] C.-Y. Hsu. *Graph-Based Approach for Solving Geometric Constraint Problems.* PhD thesis, Department of Computer Science. The University of Utah, June 1996.

[13] R. Joan-Arinyo and A. Soto. A rule-constructive geometric constraint solver. Technical Report LSI-95-25-R, Department LSI, Universitat Politècnica de Catalunya, 1995.

[14] R. Joan-Arinyo and A. Soto. A set of rules for a constructive geometric constraint solver. Technical Report LSI-95-19-R, Department LiSI, Universitat Politècnica de Catalunya, 1995.

[15] R. Joan-Arinyo and A. Soto. A correct rule-based geometric constraint solver. *Computer & Graphics*, 21(5), 1997. To appear.

[16] R. Joan-Arinyo and A. Soto. A ruler-and-compass geometric constraint solver. In M.J. Pratt, R.D. Sriram, and M.J. Wozny, editors, *Product Modeling for Computer Integrated Design and Manufacture*, pages 384 – 393. Chapman and Hall, London, May, 18-23 1997.

[17] G. Laman. On graphs and rigidity of plane skeletal structures. *Journal of Engineering Mathematics*, 4(4):331–340, October 1970.

[18] L. Lovász and M.D. Plummer. *Matching Theory.* Number 29 in Annals of Discrete Mathematics. North-Holland, 1986.

[19] N. Mata. Solving incidence and tangency constraints in 2D. Technical Report LSI-97-3R, Department LiSI, Universitat Politècnica de Catalunya, 1997.

[20] K. Murota. *Systems Analysis by Graphs and Matroids.* Algorithms and Combinatorics 3. Springer-Verlag, 1987.

[21] B.K. Rosen. Tree-manipulating systems and Church-Rosser theorems. *Journal of the ACM*, 20(1):160–187, January 1973.

[22] A. Verroust, F. Schonek, and D. Roller. Rule-oriented method for parameterized computer-aided design. *Computer Aided Design*, 24(10):531–540, October 1992.