Qualitative Study of Geometric Constraints

Hervé Lamure, Dominique Michelucci

Ecole des Mines
158 Cours Fauriel
F 42023 Saint-Etienne Cedex 02
e-mail: lamure@emse.fr, micheluc@emse.fr

Abstract

Modelling by geometric constraints is a promising method in the field of CAD/CAM. However, this process, closely related to computer programming, is also prone error. A geometric constraints based modeller should help the end user to find his mistakes , or, better, not to commit ones by watching the building process. The well known main cause of errors with these methods is the specification of redundant constraints, and sometimes conflicting constraints. It's also important to detect under-constrained parts of the system involving indecisiveness. This paper alludes to some usual numerical and probabilistic tools that could be used for this goal. There's also a decomposition method of constraints systems using tools of the graph theory. Since these two approaches have their own advantages it's worth combining them. All these methods will be studied first for systems of equations and then extended to the more specific case, but also more interesting for modelling, of systems of constraints.

1 INTRODUCTION

In CAD-systems, *Geometric Modelling by Constraints*, also called *Variational Modelling*, enables designers to describe shapes by interactively editing a sketch and specifying geometric or engineering constraints: see R. Anderl & R. Mendgen for a survey [AM95].

Engineering constraints typically involve material properties or manufacturing parameters, and are best expressed by equations. Geometric constraints fundamentally involve points, lines, circles: typical ones specify the distance between two geometric elements (two points, a point and a line, two parallel lines), the angle or the parallelism between two lines, the tangency between a line and a circle or between two circles, the incidency between a point and a line or a circle, etc. Actually, any algebraic equation involving coordinates is a possible constraint, as far as it is independent of the used coordinates system (this is detailed in section 1.3.

The usual geometric constraints can be represented either by algebraic equations, or by predicates. The predicates formulation has the disadvantage to restrict the set of expressible geometric constraints, and to prevent the merging of geometric constraints and of engineering ones, but it seemed more suited to rule-based approaches than the equational formulation. Up to now, all rule-based or qualitative approaches (as opposed to numerical methods) [Bru86, VSR92] rely on the predicates formulation. In contrast, this paper will show that the equational formulation also permits a qualitative study as well.

1.1 Solving Systems of Constraints

Constraints-based modellers have to solve in some way the set of constraints by one of the following approaches: symbolic, numerical or decomposition methods.

Symbolic methods resort to tools from Computer Algebra, like resultants or Grobner Bases [AM95]. Due to their exponential running-time, they can be used only for small systems.

Numerical methods typically improve the initial and rough guess interactively provided by the user during the sketching stage, by using some Newton iterations [AM95], or the homotopy method [LM95] which has a more intuitive convergence.

Decomposition methods reduce constraint systems into basic problems, the solutions of which are then stuck back together [Owe91, VSR92, BFH$^+$95, But79]. In $2D$ basic problems are triangles: the relative location of their 3 vertices are determined by 3 constraints (*e.g.* either 3 distances or 2 distances + 1 angle or 1 distance + 2 angles); quadrilaterals or parallelograms [VSR92]; or other systems soluble by 'ruler and compass' like Appolonius's problem. The idea is that the basic problems can be solved by applying a simple mathematical formula, and the merging stage only needs some displacement. The decomposition may be performed either by an inference engine firing rules or Prolog predicates [But79, VSR92], or by searches in the graph of constraints [BFH$^+$95, Owe91]. Up to now, this method seems restricted to $2D$ or $2D\frac{1}{2}$ applications. Moreover it often assumes that the graph of constraints verify some strong assumptions, for instance being biconnected [Owe91], or hierarchically reducible into triangles [BFH$^+$95], or that all geometric elements have exactly 2 degrees of freedom (*ie* the radius of all circles must be known), or that each constraint involves exactly 2 geometric elements. The proposed extensions to overcome some of these limitations are not natural.

1.2 The Need for a Qualitative Study

Specifying constraints is a rather abstract task and so a very error-prone process, like programming. To be truly user-friendly, it is crucial for geometric modellers by constraints to help their users to debug and tune their systems of constraints: It is not enough that the resolution stage detects some problem and informs the user that "there is a problem somewhere in his system of 100 equations and 100 unknowns"; the user wants a precise diagnosis like: "Among these 100 equations, these 3 are redundant and you can remove anyone of them; on the other hand these 3 unknowns are under-constrained: one equation is missing". As an aside, it is the reason why we are a little reluctant about using an optimization scheme to solve constraints: there is always a solution! and it is much more difficult for the user –and for the modeller– to find why the solution is not the expected one and where the mistake is.

Moreover some qualitative studies of systems of equations permit to decompose systems of constraints into simpler ones. Not only does it speed up the resolution, but it can also make it possible to solve a problem that would not be solved otherwise. Sometimes, this decomposition can also help the user to improve his understanding of the properties of his problem.

1.3 Systems of Equations VS Systems of Constraints

A well behaved *system of equations* is supposed to determine n unknowns by n independent equations.

Systems of constraints met in Geometric Modelling by Constraints are special systems of equations, since well behaved systems of constraints: distances, angles, tangencies,... between geometrical elements, or engineering equations, are *independent of the used system of coordinates* (assuming it is orthonormal), and thus determine unknown coordinates of points only up to a displacement in space. For instance, in a well behaved system of constraints in $2D$, three equations are missing to completely determine unknown coordinates of points, since a $2D$ displacement is specified by three numbers: say a translation relatively to x and y axis and a rotation around the origin. In $3D$, six equations are missing, and $d(d+1)/2$ in $\mathbb{R}^d$.

To avoid ambiguities, we will use the terms : "well-constrained", "under-constrained" and "over-constrained" only for systems of equations. We will use the terms: "rigid", "under-rigid" and "over-rigid" for systems of geometric constraints.

What if one wants to specify a constraint depending on the coordinates system, say if one wants to specify the abscissa of a point A ? The equation $A_x = v$ clearly depends on the coordinates system... The solution is to explicitly represent the coordinates system Oxy by three vertices: O, X, Y, and to specify the three constraints: $||OX|| = 1$, $||OY|| = 1$ and $\text{angle}(OX, OY) = \frac{\pi}{2}$. Then $A_x = v$ can be represented by a licit geometric constraint on the (signed) distance between OY and A: $\text{dist}(OY, A) = v$.

We will impose a last restriction on geometric constraints. To solve over-constrained systems, or possibly over-constrained systems: $f_i(x) = 0$, some people try to minimize or to make vanish the sum of the squares of the equations: $\sum_{i=1}^n f_i(x)^2$ (we think it is not a so good idea, but it is not the concern here). If the $f_i = 0$ are independent on the coordinates system, so is $\sum_{i=1}^n f_i(x)^2 = 0$. However we will forbid the use of such a trick. Or in other words, our methods are abused by this trick.

1.4 Content of the Paper

This paper will propose two methods for the qualitative study of systems of equations and for systems of constraints: the first method is a probabilistic numerical one, the second one stems from graph theory and does not consider the numeric details of equations, or geometric constraints. So it can be applied to geometric constraints represented by predicates (in opposition to equations), say Prolog predicates like in Brüderlin's works [Bru86].

Section 2 presents the probabilistic and numerical method. It is first applied to systems of equations and then extended to systems of geometric constraints. Section 3 presents the graph-based approach, similarly it is first applied to systems of equations and then extended to systems of constraints, a detailed example illustrates the process. Section 4 presents possible extensions and improvements.

We only consider systems of equations or constraints, and not inequalities, because the qualitative study of such systems is too much difficult: from elimination theory, solving systems of algebraic equations is simply exponential, whereas solving systems of equations and inequalities is doubly exponential [HM93]. Moreover, inequalities are the more often used only to select the wanted solution in the finite but very big solution set of a system *without* inequalities: so we can at least diagnosis this last system.

2 THE NUMERICAL PROBABILISTIC METHOD

2.1 The Qualitative Study of Systems of Equations

For the qualitative study of systems of equations to be feasible, some "genericity hypothesis" are needed. The weaker one is always satisfied in our context, where equations are typically algebraic ones, with real coefficients: exactly n independent equations are needed to determine n unknowns in $\mathbb{C}^n$. Actually, since we are interested only by real solutions and though it is a bit cavalier, we will assume a little stronger hypothesis and replace $\mathbb{C}^n$ by $\mathbb{R}^n$, in spite of some algebraic singular systems like for instance $x^2 + y^2 = 0$ which has a finite number of real solutions though there is only 1 equation for 2 unknowns. More generally, we assume that the trick of taking the sum of the squares of some equations will not be used. Some of the methods presented below need stronger genericity hypothesis, given later in section 3.1.3.

By the genericity hypothesis, a system in n unknown and n equations: $E(X) = 0$ where $X = (x_1 \dots x_n)$ and $E = (e_1 \dots e_n)$, is incorrect iff the jacobian $|E'| = |(\frac{\partial e_i}{\partial x_j})|$ is identically null. Mathematically speaking, things are simple. However, the symbolic computation of the determinant of the jacobian is impraticable, even for little values of n like 10: the determinant has an exponential number of terms.

A solution is to use the probabilistic scheme [Mar71, Sch80]: the value of the jacobian is computed at some random sampling points (three or four points is enough). If each time it vanishes, then there's a probability very close to 1 that the jacobian is identically zero (see previous references to have the probability of errors). Otherwise –and obviously!– it is not identically zero.

To avoid inaccuracy problems, this test can be performed for integer points and modulo some prime integer, big enough (say about 10^6 or 10^9) to decrease the risk of unlucky reductions: there is an unlucky reduction when the value of the determinant is a non null mutiple of the used prime. For relevance, real coefficients must be represented in an exact way, for instance $\sqrt{2}$ must be represented by an auxiliary unknown α and an equation $\alpha^2 - 2 = 0$.

We can even do better and compute (always probabilistically, for some random sample points) the rank or a base of the jacobian. Before we detail this, let us get a more intuitive insight on their meaning: we consider the unknowns $X = (x_1 \dots x_n)$ as a function of the time, t; deriving $E(X(t)) = 0$ relatively to time, we obtain $\dot{X}E' = \dot{0}$, where $\dot{X}$ is the vector of velocities of X, compatible with E. If E is correct, *ie* if all equations e_i are independent so E' has maximal rank n, then the only solution for $\dot{X}$ is $\dot{0}$: it is impossible to move the unknowns and to keep $E(X) = 0$. More generally, the solution for $\dot{X}E' = \dot{0}$ form a vectorial space: the kernel of E' which is dual to the space spanned by the equations e_i. The rank of the vectorial space $\dot{X}$ is the number of missing determinations. The number of equations minus the rank of E gives the number of redundant equations in E.

It is worth computing a base of $\dot{X}$ and E. It can be done in $O(n^3)$ time by several standard techniques of linear algebra (say by the Gram-Schmidt's orthogonalization process). An unknown x_k is fixed by the system E iff $\dot{x}_k$ is zero for all the vectors in the base of $\dot{X}$; otherwise the number of vectors with a non zero component $\dot{x}_k$ in the base for $\dot{X}$ gives the number of remaining degrees of freedom for x_k. Of course, the numerical values of vectors, for bases of $\dot{X}$ and E, depend on the random sampling points, but the structure of E' (rank, corank, fixed and not fixed unknowns) does not.

Last but not least, this approach (probabilistically computing a base for $\dot{X}$ and E) can also be used when the number of equations is not equal to the number of unknowns. Thus it can also be performed on-line: in an interactive context, equations are likely introduced once at a time. Thus it is possible to maintain an orthogonal base of the equation-vectors e_i, for each random sampling point. When a new constraint e is introduced, e is decomposed into $e = e_p + e'$ (by Gram-Schmidt's orthogonalization process) where e_p is the projection of e in the range of the base and e' the orthogonal part. If e is independent ($e' \neq 0$), e' is inserted in the base; otherwise the software proposes the user to remove one of the redundant constraints among the minimal redundant subset. If needed, a base of the compatible velocities $\dot{x}_j$ can also be maintained on-line.

This method is simple and very easy to implement. It may use the sparsity of the system at hand in order to speed up computations. Anyway it cannot be slower than the numerical resolution method. It can distinguish between dependent and independent subsets of equations on one part, and between fixed and unfixed unknowns on the other part. But it cannot distinguish between fixed but well-constrained unknowns, and fixed but over-constrained ones; this notion is relevant only for sparse systems: for instance in the system $x + y + z = 2x + 2y + 2z = 0$, unknowns x, y and z are in the same time over-constrained (since they are involved by redundant equations) and under-constrained (since compatible velocities $\dot{x}$, $\dot{y}$ and $\dot{z}$ are not zero).

2.2 The Qualitative Study of Systems of Constraints

Since the well behaved systems of constraints are under-constrained systems of equations (as already explained in section 1.3), this method does not directly apply to systems of geometric constraints. A straightforward way to overcome this limitation is to add some adhoc equations. If for instance in $2D$, specify for some couple of points A and B that $A_x = A_y = 0$ and $B_y = 0$. In $3D$ specify that for some triplet of points A, B and C that $A_x = A_y = A_z = 0$, $B_y = B_z = 0$ and $C_z = 0$. Then all the numerical probabilistic approach directly applies: it is possible to know the fixed and not fixed unknowns, and to detect dependent subsets of equations.

The ergonomical issues are not our concern here, but in passing we propose the following visual protocol for helping the user to see which part of his sketch is still under-determined. Let X_1 be the current solution of the set of constraints $F(X) = 0$. If some part of the sketch is under-constrained, then there is some velocity vector $\dot{X}_1$ such that $\dot{X}_1 F'(X_1) = 0$. The idea is to smoothly deform the sketch from its state X_1 to a state X_2, X_3, etc but keeping $F(X_i) = 0$ for all i. X_2 is computed from $X_1 + \epsilon \dot{X}_1$, with some correction by some Newton-Raphson's iterations or some gradient descent. Idem for X_3 from X_2, and so on. It is also easy to allow the user to specify the 2 motionless points A and B in $2D$: so he will see which part remains motionless relatively to AB, *ie* the maximal rigid part containing AB. The same holds in $3D$ for 3 motionless points. Of course, the modeller can also automatically deduce the maximal rigid parts, in the same way.

3 THE BIPARTITE GRAPH-BASED METHOD

An alternative approach, stemming from graph theory, exploits the sparsity of systems of equations or geometric constraints. The idea is to consider the bipartite graph associated to a given system of equations or geometric constraints: each unknown is associated to an unknown-vertex, each equation is associated to an equation-vertex, and an edge join an unknown-vertex and an equation-vertex iff the corresponding unknown appears in the corresponding equation (Fig. 1). This graph is bipartite because there is no edge between any two unknown-vertices, and between any two equation-vertices. In the following we will be drawing equation-vertices above the unknown-vertices.

We first apply this method to systems of equations in section 3.1, then extend it to systems of geometric constraints in section 3.2.

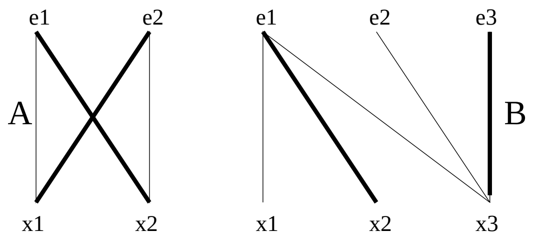

Figure 1: *A is well-constrained in generic case : it has a perfect matching. B is always singular : it has no perfect matching.*

3.1 The Qualitative Study of Systems of Equations

3.1.1 Dulmage-Mendelsohn's Decomposition

In the generic case (we will come back later to this point, see section 3.1.3), the study of this bipartite graph first permits us to decompose the system into its well-, over- and under-constrained parts [LP86].

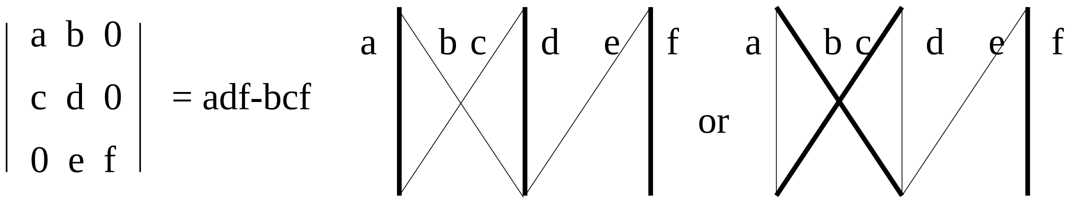

Figure 2: *A graph and its perfect matchings. Note the correspondance between each perfect matching and each term of the determinant.*

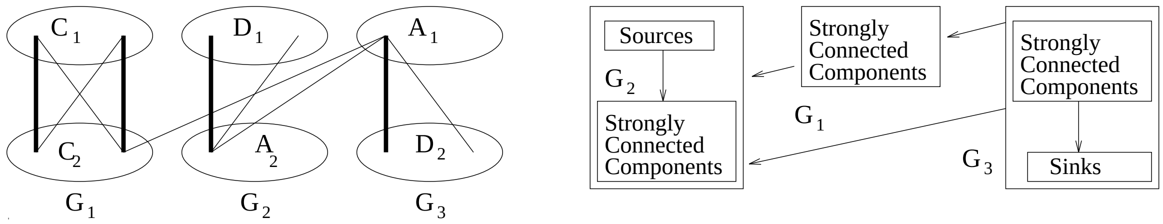

Figure 3: *The Dulmage-Mendelsohn's decomposition of a graph G, with a maximum matching, and the structure of G'.*

This decomposition is due to Dulmage and Mendelsohn. Though it was not initially developed with the study of systems of equations in mind, it appears to be relevant for this problem. We will need the following definitions:

A *matching* of a graph is a subset of its edges, such that any two distinct edges of the matching never have a common vertex. A matching is *maximum* iff it is maximal in cardinality. A vertex is *saturated* by a matching iff it is a vertex of one edge of this matching. A matching saturating all vertices of a graph is called *perfect*: see Fig. 2. From an algorithmic point of view, polynomial algorithms to compute a maximum matching for bipartite graphs are known, for instance Hopcroft and Karp's method [HK73, AHU83].

The Dulmage-Mendelsohn's decomposition is illustrated figure 3. The main properties are, in a jumble: There is no edge between D_1 and C_2, between D_2 and C_1 and between D_1 and D_2. $G_1 = C_1 \cup C_2$ has a perfect matching, so $|C_1| = |C_2|$. D_1 and D_2 are respectively the set of equation- and unknown-vertices which are not saturated by at least a maximum matching. A_2 (respectively A_1) is the set of the neighbours of D_1 (respectively D_2). Vertices of A_2 and A_1 are all saturated by all maximum matchings. Edges between C_1 and A_2, between C_2 and A_1, between A_1 and A_2 never belong to a maximum matching.

The important thing for us is that $G_1 = C_1 \cup C_2$ is the well-constrained part of the system, $G_2 = D_1 \cup A_2$ its over-constrained part, $G_3 = A_1 \cup D_2$ its under-constrained part. Only the G_1 part has a perfect matching (Fig. 2).

In order to give a fast method for computing the Dulmage-Mendelsohn's decomposition, we need the following definitions: a *directed* graph is said to be *strongly connected* iff for any pair x and y of vertices there exists a *directed path* from x to y and from y to x. The *strongly connected components* of a graph are its maximal strongly connected subgraphs, they partition its vertices. *Strongly* connected components must not be confused with connected components: the latter does not take into account the orientation of edges.

Let M be any maximum matching of the bipartite graph G. G' is the *directed* graph obtained from G

by replacing each edge (x, y) in M by two arcs xy and yx, and by orienting all other edges from equation-vertices to unknown vertices. The *strongly connected components* of G' are included either in G_1, or in G_2, or in G_3. Moreover if there are non saturated equation-vertices, then they are the sources of G_2. Symmetrically, if there are non saturated unknown-vertices, then they are sinks of G_3. Thus G' has the structure shown in figure 3. An algorithm to compute the Dulmage-Mendelsohn's decomposition follows: Find a maximum matching M of G; build the directed graph G' from G; G_2 is the set of all descendants of sources of G'; symmetrically G_3 is the set of all ancestors of sinks of G'; finally $G_1 = G' - G_2 - G_2$. Of course, this decomposition is unique and does not depend on the initial maximum matching. Finding a maximum matching can be done in $O(e\sqrt{v})$, where e is the number of edges of G and v its number of vertices, by using Hopcroft and Karp's method [HK73, AHU83]. The other steps can be done in $O(e+v)$: the computation of ancestors and descendants is made by a classical depth first or breadth first search in linear time. Clearly $e = O(v^2)$. In practice, e is the more often proportional to v. Anyway, we have a fast method to diagnose a system of equations.

3.1.2 Finding the Irreducible Subsystems

Secondly, for well-constrained systems, the study of the associated bipartite graph permits to find its irreducible subsystems and the dependencies between them. This reduction greatly speeds up the resolution process, whatever the resolution method used, numerical or symbolic. Moreover, if the use of this decomposition is visible for the user, it allows him to follow the resolution process step by step, *ie* irreducible after irreducible: the decomposition gives the running trace of the resolution process and makes it more self explanatory.

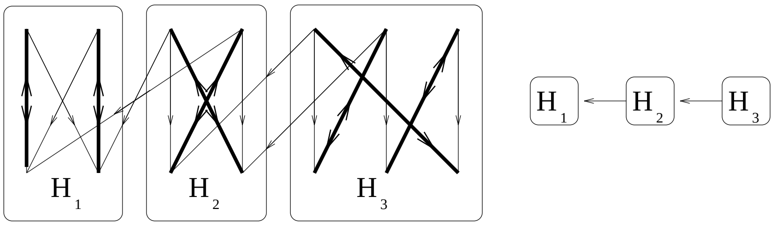

Figure 4: *Graph G', and its reduced graph.*

Let G the graph of a well-constrained system, thus $G = G_1$, $G_2 = G_3 = \emptyset$ and G has a perfect matching (in passing, it is König-Hall's theorem). An irreducible subgraph of G is a minimal subset of k equation-vertices and *all* their k adjacent unknown-vertices: it corresponds to a minimal subsystem of k equations involving k unknowns, which can be solved independently. Figure 4 shows an example of a well-constrained graph G with an irreducible subgraph H_1; $G - H_1$ has an irreducible subgraph H_2; The remaining graph $G - H_1 - H_2$ is itself irreducible. On this example (see Fig. 4), one can see that H_1, H_2 and $H_3 = G - H_1 - H_2$ are just the *strongly* connected components of the oriented graph G', obtained from G in the same way than in the previous section. Actually, it is a general property [LP86], which gives us a fast method to compute the irreducible subsystems of a given system: a perfect matching M of G is first computed; the directed graph G' is built as before; the *strongly* connected components of G' are computed, say by Tarjan's linear-time method [Tar72, AHU83]: they are the searched subsystems, and they do not depend on the initial perfect matching. To obtain the dependencies between the irreducible subsystems, build the reduced graph R from G' by contracting each *strongly* connected component in a vertex (Fig. 4): each remaining arc in R, say from s_1 to s_2, means that subsystem s_1 uses some unknowns of s_2, thus s_2 has to be solved before s_1. Of course R is acyclic and defines a partial order between the subsystems. A compatible total order between subsystems can be obtained by any topological sorting of R, or as a byproduct of Tarjan's method: it is the order in which the *strongly* connected components are obtained. All steps are linear, except the determination of a starting matching, in $O(e\sqrt{v})$. This decomposition has been already published [AAJM93]. In practice, the time for the decomposition is always negligible in front of the resolution time. For reducible systems, speed up factors of 10 or more are usual.

3.1.3 Genericity Hypothesis

The graph-based approach uses only the structure of the system and forgets all numerical informations. Thus it assumes stronger genericity hypothesis than the numerical probabilistic approach. For instance the graph A in Fig. 1 is associated to singular systems, in degenerate cases (like: $x + y = 2$, $2x + 2y = 4$ which has an infinity of solutions or like $x + y = 2$, $2x + 2y = 3$ which has no solution at all), and also to well-constrained systems, in the generic case. The probabilistic approach can discriminate between the non-generic and generic cases, but the graph-based approach obviously cannot. Note that the nullity of the jacobian in degenerate cases has nothing to do with the structure of the system, and that it suffices to randomly and infinitesimally perturb the coefficients of such degenerate systems (without changing their structure) to recover the generic situation; or, in other words, degenerate cases occur with probability 0 [LP86]. On the other hand, when the graph-based approach states that a system is not well-constrained, then it is not (see graph B in Fig. 1) and no matter the values of the coefficients are. Mathematically speaking, a sufficient condition for a system to be *generic* is that its coefficients are algebraically independent (of course it is a very unrealistic assumption, since coefficients are integers or floating point numbers, *ie* rationnal numbers ...) so that they do not verify any parasitic condition, and the rank of the jacobian is then strictly equal to the cardinality of the maximum matching of the corresponding bipartite graph. Degeneracies can only decrease the rank.

3.2 The Qualitative Study of Systems of Constraints

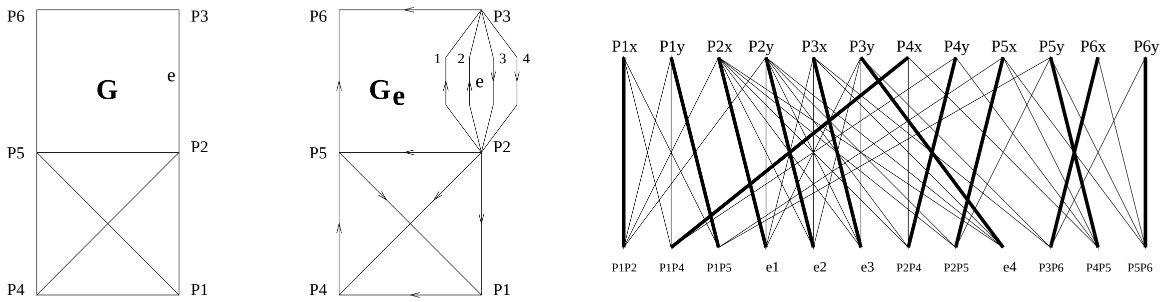

Figure 5: *This graph G is incorrect, ie non rigid. However, adding some three adhoc equations (the quadrupled edge: e) gives a graph G_e, whose associated bipartite graph $Bip(G_e)$ is well-constrained. The perfect matching of $Bip(G_e)$ is drawn directly on G_e by orienting its edges.*

The extension from system of equations to systems of geometric constraints is not simple for the bipartite graph-based approach. There are cases where adding some adhoc equations produces a bipartite graph having a perfect matching, *ie* indicating that the system is rigid, though it is not. Such a case is illustrated in $2D$ figure 5: the vertices of this graph of constraints represent, say, points with unknown vertices, and edges represent constraints of distance. In Figure 5, one of the original constraints has been quadrupled: we pin the corresponding edge e on the plane. Call G_e the resulting graph. The bipartite graph $Bip(G_e)$ has a perfect matching, though the initial system of constraints is obviously incorrect: the square with its two diagonals is over-rigid, the other square is under-rigid.

The perfect matching has been displayed in a very compact way, directly on the graph as an orientation of its edges: an edge (a, b) in a graph of constraints G_e is oriented from a to b iff, in $Bip(G_e)$, the edge (ab, a_x) or (ab, a_y) belongs to the matching. The fact that the matching is perfect implies that each vertex in G_e receives exactly 2 arcs.

However, pinning another edge of the graph in Fig. 5 permits to detect that this graph is over-rigid in one part and under-rigid in the other part. We just have to pin each and every of its edge. We now see a general formulation of the method.

3.2.1 Decomposition algorithm of systems of contraints

It is possible to use bipartite graphs to study $2D$ systems of constraints.

Hypothesis For the sake of simplicity, we assume that the system of constraints involves n unknown $2D$ points, the coordinates (x_i, y_i) of which depend on the coordinates system, and k unknowns $u_1 \ldots u_k$ which are independent of the coordinates system: they can be (possibly signed) distances, (possibly signed) areas, scalar products, angles, radii of circles ... or non geometric unknowns.

Moreover, all constraints must be independent of the coordinates system: if (x_i, y_i, u_j) is a solution, so is $(x_i + T_x, y_i, u_j)$ for any value T_x (a translation in x has been applied), so is $(x_i, y_i + T_y, u_j)$ for any value T_y (a translation in y has been applied), and so is $(x_i \cos\theta - y_i \sin\theta, x_i \sin\theta + y_i \cos\theta, u_j)$ for any value of θ (a rotation around the origin has been applied). Such a constraint either involves no points, or involves at least 2 points, but never a single point. Note that we accept contraints involving more than two points: these systems are more general than the graph of constraints illustrated by figure 5 or 6, where constraints must involve exactly 2 geometric elements (since an edge has 2 vertices). According to this criteria the bipartite graph-based approach has a larger expressive power than the one relying on graphs of constraints [Owe91, BFH+95].

Such a system of constraints has $P(n, k) = 2n + k - 3$ degrees of freedom when $n \geq 2$ and $P(n, k) = k$ degrees when $n = 0$. To fix without redundancy the $2n + k$ unknowns up to a displacement in the plane, in other words to be rigid, the system must have exactly $P(n, k)$ equations (which is trivial to check) and all its subsystems with n' points and k' other unknowns must have at most $P(n', k')$ equations: if one has more than $P(n', k')$ equations, it is over-rigid. For short, we will say that a rigid system must have the P property.

We assume that the system at hand has $P(n, k)$ equations, and that its Dulmage-Mendelsohn's decomposition has an empty part G_3 (the over-constrained part): in other words, all equations are covered by a maximal matching. Now, by slightly extending a method due to Hendrickson [Hen92], it is possible to verify that the system fullfills P, *ie* that there is no over-determined points:

Decomposition For each constraint e involving at least one point (and so, at least two points as we have just seen) in the system G, consider the bipartite graph $Bip(G_e)$ where G_e is obtained from G by adding three adhoc equations involving the same unknowns than e: intuitively speaking, we "pin" two points appearing in e on the plane. Verify that $Bip(G_e)$ has a perfect matching: if not, the system G_e is incorrect and the Dulmage-Mendelsohn's decomposition gives an over-constrained part in G_e, and so an over-rigid part in the initial system of constraints G. Hendrickson has proved [Hen92] that the system G fullfills P iff all the bipartite graphs G_e have a perfect matching: the proof is not difficult but a bit lengthy and must be omitted for conciseness.

Only the first maximum matching has to be computed from scratch: the others can be obtained more quickly, in linear time, by updating the previous one. For this reason, this method may be used on-line, when constraints are added or removed one at a time, but this point is not detailed due to lack of space. Assuming that $k = O(n)$, and that each constraint involves $O(1)$ unknowns (which is generally the case), then the first step could be done in $O(n^2)$, and each of the n other steps could be done in $O(n)$, so this method will work in $O(n^2)$, whereas the probabilistic numerical method is in $O(n^3)$.

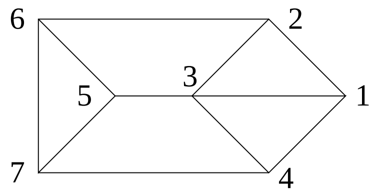

Figure 6: *A graph of constraints.*

Moreover, at each step (*ie* for each G_e), it is possible to compute the decomposition into irreducible parts: each irreducible part – it contains the particularized constraint – gives a rigid subsystem, assuming the system is correct. For instance, for the system of constraints in Fig. 6, when pinning edge-constraint $(1,3)$, subsystems $\{1,3\}$, $\{1,2,3\}$ and $\{1,3,4\}$ are found. When pinning edge-constraint $(5,6)$ or $(6,7)$ or $(5,7)$, subsystem $\{5,6,7\}$ is found (and the trivial subsystem equal to the pinned edge itself, of course). Thus *all* rigid subsystems are found this way. If the subrigid parts are "small" (*ie* involve only a number of other subrigids independent of n) then we can use this decomposition in order to speed up the resolution process. Indeed it's possible to solve each of the small systems in constant time: by applying a resolution scheme in simple cases (for example when only 3 equations are involved), by rewriting equations, or by formal resolution [DMS97].

We have used the obvious fact that a rigid system must have the P property; when the constraints are solely distances between vertices, G. Laman [Lam70] has been able to prove the converse: a system verifying the P property is rigid, so the P property is a full characterization of these rigid graphs. But as far as we know, this converse has not been extended up to now for any kind of $2D$ constraints.

As already seen in section 4.1, and for the same reasons, it is worth combining the graph-based approach and the numerical probabilistic one.

3.2.2 An Example

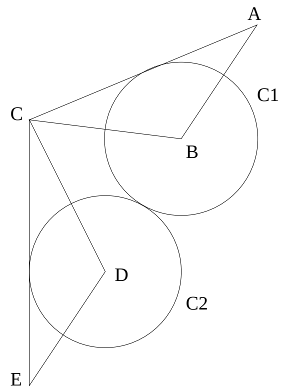

Figure 7: *A $2D$ system of geometric constraints.*

This section presents a simple and complete $2D$ example: see figure 7. The constraints are:

- $\alpha^2 - 2 = 0$ which yields $e_0(\alpha) = 0$
- $||AC|| = 4\alpha$ which yields $e_1(A, C, \alpha) = 0$
- $||AB|| = 3\alpha$ which yields $e_2(A, B, \alpha) = 0$
- $||BC|| = 2\alpha$ which yields $e_3(B, C, \alpha) = 0$
- the circle C_1 with center B and radius R_1 is tangent to the line (AC) which yields $e_4(A, B, C, R_1) = 0$
- $||CD|| = 4$ which yields $e_5(D, C) = 0$
- $||CE|| = 3$ which yields $e_6(E, C) = 0$
- $||DE|| = 2$ which yields $e_7(D, E) = 0$
- the circle C_2 with center D and radius R_2 is tangent to the line (CE) which yields $e_8(C, D, E, R_2) = 0$
- the circles C_1 and C_2 are tangent each other which yields $e_9(B, D, R_1, R_2) = 0$

The method detects that the system is rigid and its rigid subsystems are: $S_0 = \{e_0\}$ which determines α, $S_1 = S_0 \cup \{e_1, e_2, e_3\}$ which determines the triangle (A, B, C), $S_2 = \{e_5, e_6, e_7\}$ which determines the triangle (C, D, E), $S_3 = S_1 \cup \{e_4\}$ which determines the radius R_1, $S_4 = S_2 \cup \{e_8\}$ which determines the radius R_2, $S = S_3 \cup S_4 \cup \{e_9\}$ which permits to assemble the 2 parts of the figure. For simplicity we have omitted the trivial subsystems $\{e_1\}$, $\{e_2\}$, $\{e_3\}$, $\{e_5\}$, $\{e_6\}$, $\{e_7\}$.

4 POSSIBLE EXTENSIONS

4.1 Combining Numerical and Graph Methods

It is worth combining the two approaches. The graph-based approach gives a decomposition into over-, under- or well-constrained parts, or into irreducible well-constrained parts, which the probabilistic and numerical method cannot produce. However these parts cannot be studied further by the graph-based approach.

Inside each irreducible and well-constrained part, the numerical probabilistic method may then be used, to detect redundancies between equations or not fixed unknown(s) due to degeneracies (probably caused by some mistakes from the user) that could not be detected by the graph-based approach: for instance when the user has specified two times the same constraint, or when he has specified 3 angles between 3 lines in $2D$, or, more generally, when a constraint either is a consequence of the others, or contradicts the others. Recall the graph-based approach can be abused by non generic systems, which will not abuse the probabilistic and numerical method[1] which uses a much weaker genericity hypothesis.

The fact that the graph-based approach is easierly abused than the probabilistic one must not be thought as a defect of the former: on the contrary, the fact that a system is found well-constrained for the former approach, and bad-constrained for the latter one is very informative, when debugging a set of constraints.

[1] except exceedingly ill luck: in practice, this situation may be neglected

4.2 The $3D$ Case

We have not investigated the use of bipartite graphs for $3D$ systems of constraints so far. The function $P(n,k)$ becomes in $3D$: $P(0,k) = k$, $P(2,\, k) = 1 + k$ and $P(n,k) = 3n - 6 + k$ when $n \geq 3$. A system with n $3D$ points and k other unknowns must have exactly $P(n,k)$ equations (which is trivial to check) and all its subsystems with n' points and k' other unknowns must have at most $P(n',k')$ equations: if one has more than $P(n',k')$ equations, it is over-determined. Hendrickson's method has to be modified in this way: for each couple of constraints involving at least a common point, we must add 6 constraints, say 3 copies of each or: $A_x = A_y = A_z = B_y = B_z = C_z = 0$, in order to "pin" in the space 3 of the points involved by the couple of constraints, and then we verify that the associated bipartite graph has a perfect matching. We have this time to consider couples of constraints, because we have to pin 3 points and a single constraint may involve only two points. The main idea is that an over-rigid subsystem, having n' points, k' unknowns and more than $P(n',k')$ constraints, contains at least such a couple of constraints; after copying each of them 3 times, we will obtain a partly over-constrained system of equations, which has not a perfect matching. Conversely, if the system fullfills P, then all bipartite graphs will have a perfect matching.

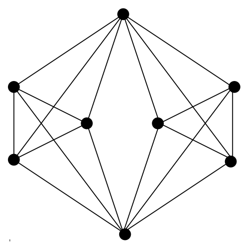

Figure 8: *Here is the classical counterexample of G. Laman's proposition in $3D$. This graph fullfills the P property, but it is not rigid. On one hand, there is no reason for the height of the left half to be equal to the height of the right half, and on the other hand, the two halves can freely rotate around the vertical axis of symmetry.*

Unfortunately, the P property is not strong enough in $3D$: some graphs fullfill the P property but are not rigid! (Fig. 8 shows a classical counterexample seen in [Hen92, LP86] and others) and G. Laman's theorem does not extend to $3D$ and beyond. Thus in $3D$ the graph-based approach seems perhaps less attractive, relatively to the numerical probabilistic one which always works, and in time $O(n^3)$, for any dimension and which is straightforward to implement. However, the graph-based approach will give in a natural way the subrigid parts contrarily to the numerical probabilistic approach and deserves further study.

4.3 Choosing the Best Solution Amongst Many

After the work of C. Hoffmann, R. Paige and their students [BFH+95], identifying the good solution of a system is now considered as an important issue of this modelling scheme. There are several approaches.

With the numerical methods, the initial guess is supposed to indicate the hoped solution, at least theoretically.

Otherwise, one idea is to choose the solution that keeps better the relative location of geometric elements in the initial guess (for instance the clockwise or anti-clockwise orientation of triplets of construction points). If this heuristic method fails, the decomposition of the set of constraints into basic problems permit [BFH+95] to interactively browse the set of all solutions, considered as a tree of choices:

for instance, an equation of degree 2 has two sons, one with the positive square root, the other with the negative square root.

The last possible method is to add inequalities which reject all but the wanted solution.

There are other essential issues, *e.g.* finding good visual metaphors to show the user the constraints and which part of its sketch is under-, well- or over-constrained, etc, but they are beyond the scope of this paper.

5 CONCLUSION

This paper has presented two techniques for the qualitative study of the system of constraints: the first, the numerical and probabilistic one, is very easy to implement and very general: it works in any dimension. The second stems from graph theory, and generalizes and simplifies previous decomposition approaches. In 2D, it is easy to implement, faster than the first approach, and moreover it gives all rigid subparts of the sketch, which permit to speed up the resolution step. Anyway, it is worth combining the two approaches: the graph-based method gives a first decomposition, each part of which may then be studied further with the numerical and probabilistic technique. Further work is needed for the graph-based approach in $3D$, which has not been implemented and tested so far.

References

[AAJM93] S. Ait-Aoudia, R. Jegou, and D. Michelucci. Reduction of constraint systems. In *Compugraphic*, pages 83–92, Alvor, Portugal, 1993. Also available at *http://www.emse.fr/˜micheluc/*.

[AHU83] A. Aho, J. Hopcroft, and J. Ullman. *Data Structures and Algorithms*. Addison-Wesley Publishing Company, Reading, Mass., 1983.

[AM95] R. Anderl and R. Mendgen. Parametric design ands its impact on solid modeling applications. In *Proceedings of the Symposium on Solid Modeling Foundations and CAD/CAM Applications*, pages 1–12, 1995.

[BFH+95] W. Bouma, I. Fudos, C. Hoffmann, J. Cai, and R. Paige. Geometric constraint solver. *Computer-Aided Design*, 27(6):487–501, 1995.

[Bru86] B. Bruderlin. Constructing three-dimensional geometric objects defined by constraints. In *Interactive 3D Graphics*, pages 111–129, October 1986.

[But79] M. Buthion. Un programme qui résoud formellement des problèmes de constructions géométriques. *RAIRO Informatique*, 3(4):353–387, oct 1979.

[DMS97] J.F. Dufourd, P. Mathis, and P. Schreck. Formal resolution of geometrical constraint systems by assembling. In Christoph Hoffmann and Wim Bronsvort, editors, *Fourth Symposium on Solid Modeling and Applications*, pages 271–284. ACM press, 1997.

[Hen92] B. Hendrickson. Conditions for unique realizations. *SIAM J. Computing*, 21(1):65–84, feb 1992.

[HK73] J.E. Hopcroft and R.M. Karp. An $n^{5/2}$ algorithm for maximum matching in bipartite graphs. *SIAM J. Computing*, 2(4):225–231, 1973.

[HM93] Joos Heintz and Jacques Morgenstern. On the intrinsic complexity of elimination theory. Technical Report 1923, INRIA, 1993.

[Lam70] G. Laman. On graphs rigidity of plane skeletal structures. *J. Engineering Math.*, 4(4):331–340, Oct. 1970.

[LM95] H. Lamure and D. Michelucci. Solving constraints by homotopy. In *Proceedings of the Symposium on Solid Modeling Foundations and CAD/CAM Applications*, pages 263–269, May 1995.

[LP86] L. Lovasz and M.D. Plummer. *Matching Theory.* North-Holland, 1986.

[Mar71] W.A. Martin. Determining the equivalence of algebraic expressions by hash coding. *J. ACM*, 18(4):549–558, 1971.

[Owe91] J.C. Owen. Algebraic solution for geometry from dimensional constraints. In *Proceedings of the Symposium on Solid Modeling Foundations and CAD/CAM Applications*, pages 397–407, 1991.

[Sch80] J.T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, (27):701–717, 1980.

[Tar72] R.E. Tarjan. Depth first search and linear graph algorithms. *SIAM J. Computing*, 1, 2:146–160, 1972.

[VSR92] A. Verroust, F. Schonek, and D. Roller. Rule oriented method for parametrized computer aided design. *Computer Aided Design*, 24(3):531–540, Oct 1992.