Engineering with Computers (1994) 10: 155–161

Engineering with Computers

A Proposed Structure for a Rule-Based Description of Parametric Forms

A. J. Medland

Centre for Geometric Modelling and Design, Department of Manufacturing and Engineering Systems, Brunel University, Uxbridge, Middlesex, UK

Abstract. *Much work has been carried out on the creation of protocols for the exchange of data between CAD systems, but very little on the transfer of knowledge. The knowledge behind the development of a product is vital if modifications and adaptations are to be made within a concurrent development environment. Current CAD systems only contain a limited level of product knowledge through user-defined parametric structures. It is the underlying relationships in these programs that need to be communicated between systems, rather than the form of their individual programming languages. A generic structure is proposed that is able to handle both the current level of parametric programming and is suited to the resolution of conflict through the use of constraint modelling procedures. The structure of a parametric description is described and decomposed into its fundamental components. These components are then re-formed within a truth maintenance structure to allow a transfer protocol to be created. The derived protocol structure is then demonstrated on a number of simple examples.*

Keywords. Constraints; Geometric; Parametric; Product form

1. Introduction

The procedures necessary for the transfer of engineering data between commercial CAD systems has required a considerable effort over the past decade [1, 2]. Only now are standards for the exchange of geometric and product information being realised and the first generation of commercial implementation becoming available [3].

Such procedures have in principle the capability to transfer product knowledge, as well as engineering data. The commercially available systems, however, are limited in their abilities to handle and manipulate such knowledge. For the full description of product knowledge it is necessary to hold not only the geometry and associativities, but also the conflicting requirements of the technical specification, manufacturing and economic conditions. Such activities require an understanding of functionality, the identification of conflict and its resolution.

The most advanced commercial systems currently address the problems of associativity within the geometric representation, and provide techniques whereby various levels of parametric descriptions can be employed. Such parametrics allow the relationships between geometries entities to be described in symbolic, geometric or mathematical terms. Thus the conditions of dependency upon other geometries, those of form (parallelism, etc.) and mathematical equations, can be incorporated.

Though limited in their structures, the parametric programs provide the first level of product knowledge through their defined relationships. It is thus these relationships that need to be transferred between systems rather than the programming languages themselves. As the commercial systems have been developed independently to address differing potential markets, there are many forms of parametric programming. These, in the main, reflect the internal structures employed for the creation and manipulation of geometric entities rather than the needs of the designer.

In order to provide a means for communicating full product information, the transfer of parametric structures should be seen as a sub-set of a generic description that can address the broader issues of knowledge transfer.

2. Parametric Structures

In order to specify relationships and form within an information transfer system (rather than simply geometry and association) it is necessary to describe the underlying generic structures, rather than the form of the parametric languages currently used. All of these

Correspondence and offprint requests to: Professor A. J. Medland, Dept of Manufacturing and Engineering Systems, Brunel, The University of West London, Uxbridge, Middlesex UB8 3PH, UK.

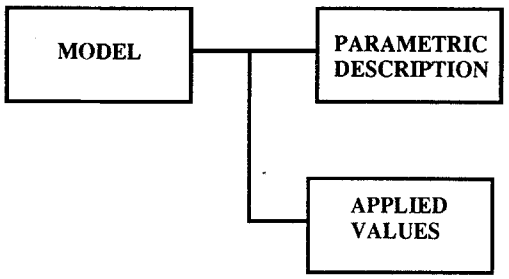


Fig. 1. High-level description of a parametric.

languages have been tailored both to reflect and support the internal structures and procedures used within a particular CAD environment. They all tend to be unique and only applicable to the one system. Even different revisions of the same software have been known to support quite different parametric languages.

The problems of transferring parametric descriptions between widely different systems are therefore considerable and can only be addressed at a fundamental level.

3. Generic Parametric Form

The structure of a parametric description can be considered in a hierarchical form. At the highest level a model is formed as a single incidence from a parametric description and a set of applied values (Fig. 1). Thus by changing the applied values, variations in the resulting model can be readily created.

In order to represent differing forms, relationships, logical conditions and constraints it is necessary to describe the parametric structure itself in greater detail. A proposed hierarchical decomposition for this is shown in Fig. 2. The form and details of the internal levels are as follows.

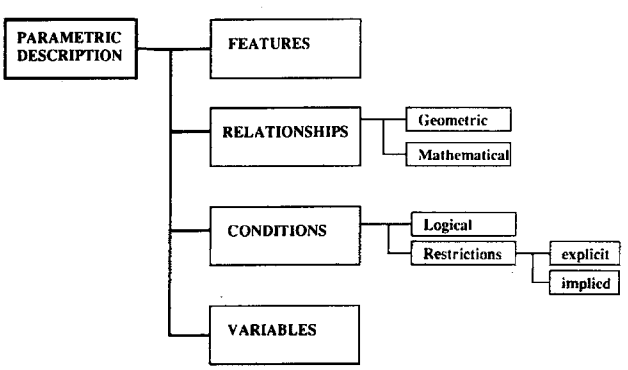


Fig. 2. Details of parametric structure.

3.1. Level 1 (Extending from the 'Parametric Description')

The first decomposition simply breaks the parametric description down into two elements, the *parametric elements* and a *sequencing list.*

In most CAD-based languages no sequencing list is necessary as the order of events in the language makes this implicit. However, the order in which the parametric elements need to be applied or resolved may change between systems or may need to be grouped or re-ordered to allow different aspects of the same problem to be addressed.

Some languages require that all parameters are declared at the commencement of the program whilst others allow any variable name to be introduced in the body of the program. Additional formal structures may be imposed to allow logical statements, file handling and looping to be included.

Whilst the program may have been constructed with a single purpose in mind, it is an advantage within a concurrent engineering environment to be able to re-order the program to allow it to be applied in a more generic way. For example, a program may be written to determine the minimum amount of material between a hole and the component edge to satisfy a given stress condition, then by changing the sequencing it is possible to find either the largest hole that can be put in a chosen position or the minimum distance to an edge for a chosen size or hole.

3.2. Level 2 (Extending from the 'Parametric Elements')

A full parametric description must allow the user to define a number of different parametric elements, These should include *geometric features, relationships, conditions* and *problem variables.*

The geometric entities, which which the final model is to be formed (in whatever representation) need firstly to be defined and held as geometric features. The relationships between these various features also need to be described. These may range from point on a line through to a Boolean operation on a combination of solids.

It may also be required that conditions are imposed on the form or validity of the model construction being proposed, by the particular selection of combinations of input values. This is detailed more fully in the description of Level 3 activities.

Finally, it is necessary to declare which of the model parameters can be changed. Normally this is held implicitly within the programming structure and is therefore different between systems. In a generic

representation this should be held explicitly in order that the various forms of representation can be re-created.

3.3. Level 3.1 (Extending from the 'Relationships')

The relationships specified within a parametric description can exist in both *geometric* and *mathematical* forms.

The model thus constructed can then draw upon or comply with existing or chosen geometric arrangements. Feature models can be built and positioned within existing models. Others can be created to conform to chosen mathematical relationships to provide function, form, strength, etc.

3.4. Level 3.2 (Extending from the 'Conditions')

Conditions imposed upon the form of a parametric representation can be both *logical* relationships and *restrictions* in geometry.

A logical qualifier can be used to determine which form of alternative actions should be taken. These may lead to variations in the parametric description through to the inclusion of completely different parametric models.

The restrictions operate directly upon the geometry of the modelling representation being used and thus work directly upon the geometric modeller (resulting in a lower description at Level 4).

3.5. Level 4 (Extending from the 'Restrictions')

The restrictions imposed upon the geometric modeller can be in an *explicit* or *implied* form, depending on the original structure of the parametric language.

Due to the order of the instructions assembled in the language, restrictions may be implied. Attachment of one entity upon another provides an implied associativity. Other implied restrictions also arise because of the variations and associations allowed within the original definition of the geometric entity.

Within a general modelling description it is necessary to define the implicit restrictions that are to be placed upon a chosen arrangement. Multiple freedoms (on terms of model translations, rotations and scales, as well as model associations) need to be restricted to provide the desired type of solution.

4. Parametric Formation

The previous hierarchical description of the parametric representation allows all elements of the parametric description to be held independently (within separate lists) and applied by use of the sequencing list to reconstruct the appropriate parametric generating activity in any parametric description. It can also, through the re-ordering of the sequencing list, be used to address differing variants of the original problem and to check the 'truth' of a proposed design. It can thus be incorporated within an optimization process to ensure compliance of the design to the chosen constraints.

The various parametric elements can be re-grouped to provide a generic schema for the creation of a solution program. Here the eight elements are formed into four entity procedures operated upon by the sequencing file (Fig. 3). The geometric and mathematical relationships together with the logical conditions and the explicit restrictions are grouped together to form the rules of the problem.

The other three procedures are formed to create the remainder of the executable program. Firstly, a constraints procedure is generated from the implied restrictions. The parametric features are assembled to create a geometry function to describe the solution. Finally, the original variables are used to create a procedure to allow these variables to be selected through the sequencing operation.

Such a schema for a parametric definition can thus be detailed as shown in Fig. 4. Here the '*sequence*' entity is used to execute a combination of activities from those of '*variables*', '*geometry*', '*constraints*' and '*rules*'. The 'variable' entity can comprise any of the declared type of parameters, whilst the 'geometry' comprises any combination of the declared geometric entities. The implied restrictions then form a set of 'constraints'.

The main analysis and resolution activity of the parametric schema is finally performed within the 'rules' entity. Here a set of rules is formed that

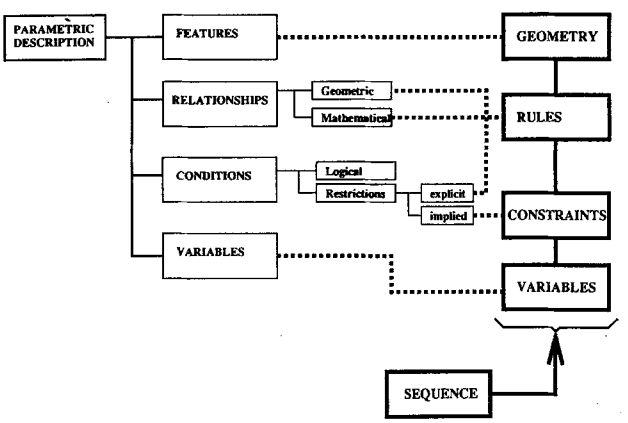


Fig. 3. Parametric grouping to form four entity procedures.

```
SCHEMA parametric;

ENTITY sequence;
  SUPERTYPE OF(ONEOF
(variables,geometry,constraints,rules));
END_ENTITY;


  ENTITY variables;
    SUBTYPE OF(sequence);
      r:  REAL;
      i:  INTEGER;
      s:  STRING;
  END_ENTITY;


  ENTITY geometry;
    SUBTYPE OF(sequence);
      p:   POINT;
      l:   LINE;
      c:   CURVE;
      ..........;
  END_ENTITY;


   ENTITY constraints;
      SUBTYPE OF(sequence);
         implied       : SET OF constraints;
   END_ENTITY;


   ENTITY rules;
      SUBTYPE OF(sequence);
      WHERE
         rule1:
         geometric     : SET OF geometric
                         expressions;
         rule2:
         mathematical  : SET OF
                          mathematical rules;
         rule3:
         logical       : SET OF logical
                         rules;

         rule4:
         explicit      : SET OF geometric
      END_RULE;          rules;
   END_ENTITY;


END_SCHEMA;
```

Fig. 4. Schema for parametric definition.

includes those created from the geometric expressions, mathematical equations, logical conditions and the rules defined by the geometric conditions. Such rules as defined by the sequencing operation can be selected and resolved by the application of the chosen 'variables'.

5. Constraint Resolution

Within a constraint modelling approach the parametric problem (or a combination of sub-problems) is resolved by the manipulation of the free variables declared for that problem [4, 5]. Thus within the schema this can be accomplished by allowing all undeclared variables within the 'variables' entity to be used as the free variables for the problem solution.

The rules are thus resolved, by direct search techniques, to a state in which all the individual rules are simultaneously true. In solving the chosen problem it is sufficient initially to define those variable entries that are necessary and the constraints that need to be applied (by sequencing firstly into the 'variables' entity and then evoking the 'constraint' entity). The complete problem is then resolved through the constraint resolution of the 'rules' entity. The resolved values are then converted and displayed in geometric form by evoking the 'geometry' entity.

New forms of the problem can then be derived simply by redefining those variables that are to be selected and those to be freed. Within such a schema for parametric descriptions, it is possible to select all variables (leaving none to be manipulated by the constraint resolution process). The technique can then establish the 'truth' of the proposed solution. If the conditions result in all of the rules being true then the solution is accepted and displayed. Should the chosen value create an untrue state then a 'SOLUTION FAILED' statement can be returned, before the failed geometric solution is displayed for consideration.

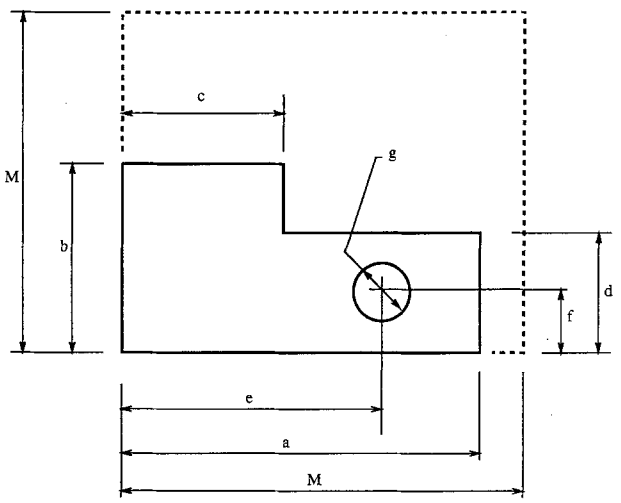
Such a parametric structure allows levels of knowledge beyond that of current parametric programs to be handled and communicated between computing systems.

6. Application of Parametric Schema

The form of the parametric schema has been applied to a number of simple problems. Here the problem definition has been created as a text file conforming to the schema structure. This file is then automatically translated into a program for the RASOR constraint modelling system. The resulting problem program is then operated (in the form defined by the sequencing instructions) by simply evoking the 'sequence' instruction. The 'constraint', 'geometry' and 'rules' functions, created during the automatic program generation, are then addressed in the appropriate manner. Upon the resolution of the problem, the solution is automatically displayed.

This approach is now illustrated by two simple examples: the first being a plate with a single hole and the second the assembly of a four bar mechanism.

The plate problem, shown in Fig. 5, is defined by



CIRCLE - to be at mid section.
- diameter to be less than d

RESTRICTION - blank size less than M by M

Fig. 5. Geometric form of plate with hole.

the parametric description shown in Fig. 6. Here the geometry, relationships, conditions and values are all described separately. In addition, a sequencing list (describing the originally intended operating order) is included at the end.

By the execution of the 'sequence' instruction all nominal values are selected and values for the maximum size of the plate blank size and hole are requested. The rules are then resolved and the geometry displayed. Such a problem state is under constrained, resulting in many valid solutions (such as shown in Fig. 7). Additional values, for any combination of plate dimensions, can be added and the 'rules' and 'geometry' functions called to create new solutions, until an untrue state is achieved by a conflict between the rules.

For the four bar mechanism problem (shown in Fig. 8), the parametric description is created as in Fig. 9. Here it is necessary to include the 'constraint' function to handle the implied conditions of pivoting and colour selection allowable within the RASOR system. The 'sequence' operation in this case selects and fixes the initially declared value for the component geometry. Only the angle of the driver crank is requested before the geometry is defined and the constraints set. The 'rules' function is then resolved and the solution displayed (as shown in Fig. 10). New values for drive angle and geometry can thus be selected by simply evoking the 'sequence' command.

```
PLATE5 43
variables 8
   real a=10 'width'
   real b=15.0 'height'
   real c=6 'width to cutout'
   real d=8 'height to cutout'
   real e=3 'hor.dist. to hole'
   real f=4 'vert.dist. to hole'
   real g=1 'hole rad.'
   real m=20 'plate size'
geometry 9
   line l1 (0,0,0,a,0,0)
   line l2 (0,0,0,b)
   line l3 (0,b,c,b)
   line l4 (c,b,c,d)
   line l5 (c,d,a,d)
   line l6 (a,d,a,0)
   line l7 (e,(f+0.5*g),e,(f-0.5*g))
   line l8 ((e-0.5*g),f,(e+0.5*g),f)
   circle c1 (e,f,(e+g),f,(e+g),f)
relationships 1
   mathematical 2
      (e - 0.5*(a+c))
      (f - 0.5*(d))
conditions 2
  logical 4
      (d le b)
      (c le a)
      (2*g le d)
      (2*g le (a-c))
   explicit 5
      (a le m)
      (b le m)
      (c ge 0)
      (d ge 0)
      (g ge 1)
sequence 7
   value "0"
   value "8"
   value "7"
   rules
   geometry
   zoom
   rpnt
```

Fig. 6. Parametric description of plate with hole.

7. Conclusions

The creation of a generic form for the description of parametric programs in CAD systems provides the opportunity to allow product knowledge to be communicated between system.

The parametric schema proposed allows all parametric elements to be described separately and accessed by a sequencing list. Within the schema the elemental descriptions can be recombined to create 'variable', 'geometry', 'constraint' and 'rule' functions that are formed into the original parametric problem by the execution of the 'sequencing' operation.

Such an approach can be used to re-form the parametric description within other parametric languages or allow its direct resolution within a constraint modelling environment.

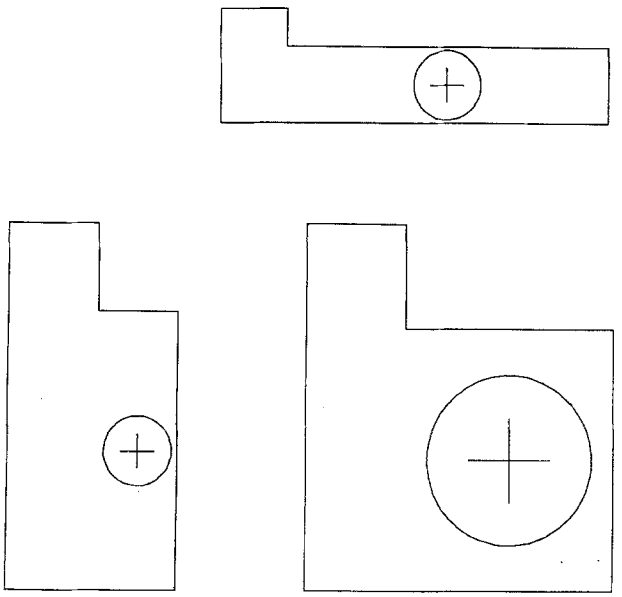


Fig. 7. Valid solutions for plate problems.

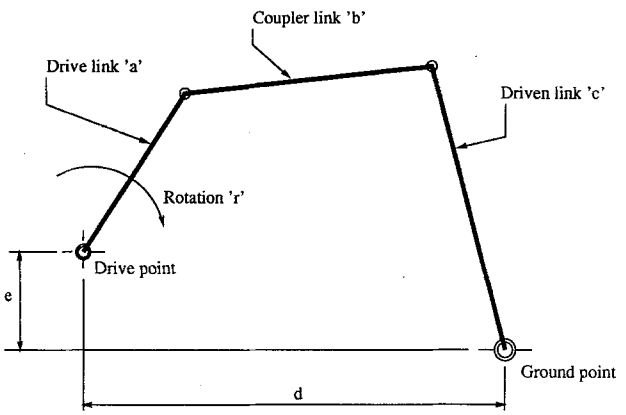


Fig. 8. Four bar mechanism geometry.

Whilst there is still a considerable amount of work necessary to make this proposed approach to parametric descriptions into a fully practical system, it has been validated upon a number of simple problems, as illustrated. More complex and real engineering problems are currently being investigated.

Such an approach, when implemented within a concurrent environment, provides the advantage of employing a generic description that can be manipulated to address intelligently a range of engineering problems. By holding separately the elements that make up the parametric description and the sequencing list, this description can be re-formed to operate within the structures of all the currently used parametric languages.

```
FOURBAR5 49
variables 6
   real a=4.0 'drive link'
   real b=10.0 'coupler link'
   real c=10.0 'driven link'
   real d=10.0 'hor.dist. to ground point'
   real e=0.0 'vert.dist. to ground piont'
   real r=0.0 'angle of rotation'
geometry 10
   model m0 (0,0,0)
   model m1 (0,0,0)
   model m2 (0,0,0,&(m1))
   model m3 (0,0,0)
   line l0 (0,0,4,0,&(m0))
   line l1 (0,0,a,0,&(m1))
   line l2 (0,0,b,0,&(m2))
   line l3 (0,0,c,0,&(m3))
   point p1 (0,0)
   point p2 (d,e)
relationships 1
   geometric 2
      (l2:e2 on l3:e2)
      (r - m0:a)
conditions 2
   implied 7
      pivot(m0, l0:e1, p1)
      pivot(m1, l1:e1, p1)
      pivot(m2, l2:e1, l1:e2)
      pivot(m3, l3:e1, p2)
      ccol(1, l1)
      ccol(2, l2)
      ccol(3, l3)
   explicit 2
      ((m1:a - m0:a) ge -5.0)
      ((m1:a - m0:a) le 5.0)
sequence 13
   value "0"
   res a
   res b
   res c
   res d
   res e
   value "6"
   geometry
   constraints
   rules
   rpnt
   zoom
   rpnt
```

Fig. 9. Parametric description of four bar mechanism.

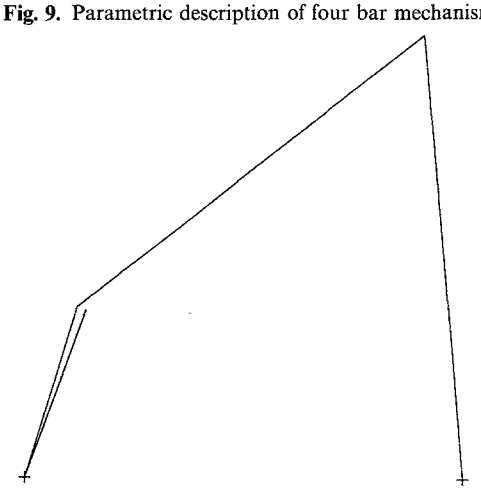


Fig. 10. Solution created from four bar mechanism parametric.

References

1. Bloor, M.S.; Owen, J. (1991) CAD/CAM product-data exchange: the next step, Computer Aided Design, 23, 4, 237–243
2. ISO 10303 (1993) STEP Standard for the Exchange of Product Data, Draft, March, ISO
3. CAD++ Programmers Reference Manual, Version 4.2 (1992) CIMIO Ltd, Egham, UK, December
4. Mullineux, G. (1988) The introduction of constraints into a graphics system, Engineering with Computers, 3, 201–205
5. Medland, A.J.; Mullineux, G. (1986) The investigation of a rule-based spatial assembly procedure, Proc. CAD86