INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN
ICED 03 STOCKHOLM, AUGUST 19-21, 2003

A GRAMMATICAL APPROACH TO COMPUTATIONAL GENERATION OF MECHANICAL CLOCK DESIGNS

Alex C Starling and Kristina Shea

Abstract

A general synthesis method based on a parallel grammar for design of mechanical systems has been improved through the use of Perturbation Rules (P-Rules) that allow parametric design changes while upholding topologic and parametric constraints. The P-Rules have been used with a generate-and-test algorithm to produce preferred mechanical clock and wristwatch designs based on various design criteria.

Keywords: computer aided design, design synthesis, functional modelling, parametric modelling

1. Introduction

Design synthesis is an area of research that involves the development of new methods and the enhancement of current techniques to improve rapid computational generation of solutions to design problems. The goal is not so much to totally automate the design process, but rather to enhance design innovation by making the computer a more effective tool for developing creative engineering design solutions [1].

Mechanical systems design covers a very large and diverse range of problems that are often ill-structured [2], where desired performance criteria cannot be easily translated into quantifiable objectives. Existing examples of the successful utilisation of computing power to help solve design problems, such as the use of optimisation methods, are often limited to tuning of design parameters and cannot generate fundamentally different types of solutions. These two issues must be addressed for developing design synthesis methods for mechanical systems. A two-pronged research strategy is therefore required to (1) enhance our ability to quantify performance of mechanical systems and (2) improve generation of new solutions.

The work presented here expands an existing computational approach for generating parametric solutions, i.e. three-dimensional design architectures, that is based on the combination of a Function-Behaviour-Structure (FBS) model [3] of the design problem and a parallel grammar [4]. Using the design of mechanical clocks as an example, the existing method enables the solution of synthesis tasks that are topologically and geometrically constrained. Work presented in this paper builds on this approach, enabling the permutation of parametric designs that have been generated. A new class of structural grammar rule upholds form and function constraints on the design and enables the search for improved parametric designs, allowing performance metrics to be used as objective functions. This is demonstrated by using direct search to find improved solutions to mechanical clock design problems.

2. Background

Design synthesis research can be traced back to the advent of the computer [5]. An overview of current state-of-the-art in design synthesis is given by Antonsson and Cagan [6]. Relevant selected examples include Chakrabarti et al. [7] who use a ‘compositional synthesis’ approach to generate solution concepts using different abstraction levels to tackle coupled mechanical design problems using exhaustive search. This work has been extended to include spatial configurations, using heuristics to ‘prune’ infeasible solutions [8]. Finger and Rinderle [9] use manually manipulated form-behaviour diagrams based on bond graphs [10] for conceptual design of topologic configurations using a part-based element library.

Sophisticated production systems have been implemented in design synthesis work. Lipson and Pollack [11] use genetic algorithms to generate robots out of struts and actuators using a fitness function based on locomotive ability. Schmidt and Cagan [12] use a graph grammar to manipulate part-based design problems, using stochastic search to evaluate designs. Li et al. [13] show how a sophisticated graph representation with a corresponding set of grammar rules can be used to generate novel designs of epicyclic gear trains. Due to their very nature, design synthesis methods are often applicable to redesign of existing products. Chase [14] outlines a model for redesign based on an FBS model using graph grammars.

The validity of designs generated by the synthesis process can be assured by the use of constraints. Bracewell and Johnson [15] have demonstrated the direct solution of a large set of design variables that are connected by various types of constraint. Szykman and Cagan [16] have investigated component packing and pipe routing problems for constraint solving by search. A multi-objective optimisation approach is used where rules for translation, rotation and swapping of objects are implemented to satisfy soft constraints.

3. Generating mechanical designs

In this work a parallel grammar [4], consisting of function and structure grammars, has been used to create parametric designs from FBS models. A build-up approach to synthesis is used where topologic and geometric constraints ensure that desired behaviour matches actual behaviour (Figure 1). Working with two separate design representations, the function and structure grammars are applied to an initial design to add and remove elements in directed graphs and 3D parametric part models respectively to create design solutions. The parallel grammar, implemented here for a clock grammar, is part of a general design synthesis framework based on a parametric synthesis methodology [17]. The parallel grammar has been implemented in C++, where function and structure representations are instantiations of two main classes. Geometric constraint satisfaction is facilitated by the use of a collision detection library, SOLID-2.0[1]; virtual prototypes of structure representations are generated using VRML.

The function and structure rules in this grammar work in parallel to create separate function and structure representations that together characterise valid design solutions. The original structure rules build up parametric structures in the design that embody a given functional representation. By virtue of their constructive nature, these structure rules are now referred to as Create Rules, or C-Rules (Figure 2). An example of how C-Rules are used to build up a parametric structure is given in Figure 3: C-Rule 1 creates an initial spindle *m*, C-Rule 2 adds

---

[1] http://www.win.tue.nl/~gino/solid/

a spindle *n* to a layout where there is an existing spindle *m* and C-Rule 3 connects these two spindles *m* and *n* via a gear pair. Iterative application of such C-Rules is used to build up a parametric structure using a generate-and-test algorithm that provides parametric data for structure elements, such as the gear disk radius values that correspond to required gear joint ratios. More detail on the C-Rules can be found in [4].

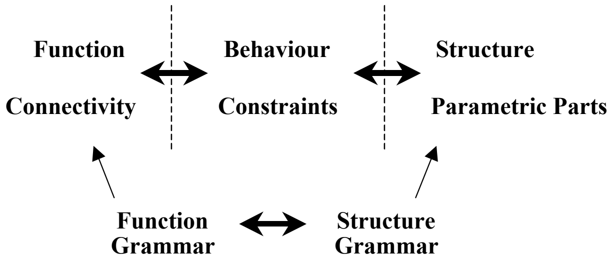


Figure 1. Parallel grammar based on FBS model.

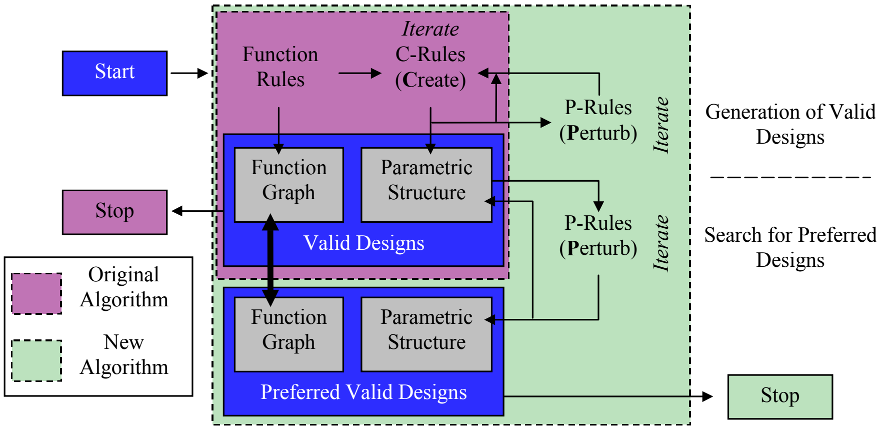


Figure 2. Framework for generation of preferred designs using the parallel grammar.

Motivated by a desire to modify structure representations generated by the existing algorithm to obtain preferred or even optimal designs, a new set of rules, called Perturbation Rules (P-Rules), have now been added to the parallel grammar to complement the existing C-Rules. These Perturbation Rules allow the variation of existing parametric structures as they facilitate parameter changes while ensuring constraint satisfaction but without altering the structural connectivity that relates to the function graph. Figure 2 shows the new extended framework for generating designs using the parallel grammar. A complete set of the P-Rules implemented for the clock grammar example are shown in Figure 4.

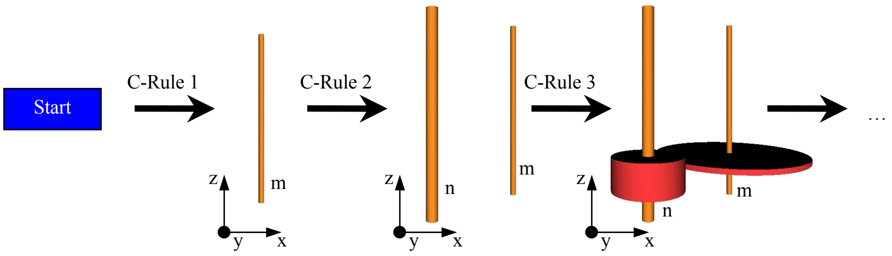


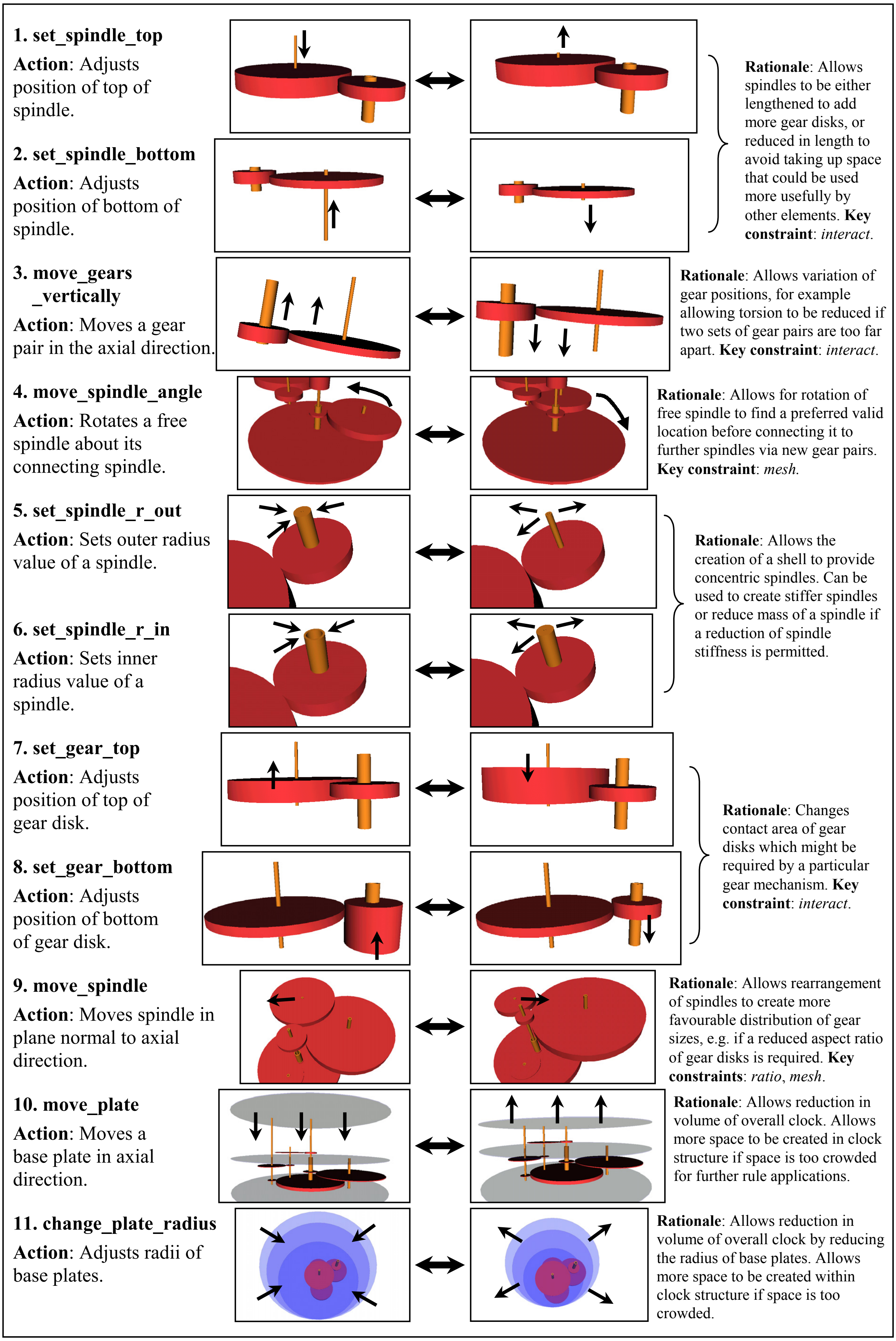
Figure 3. Application of C-Rules to build up parametric parts. Spindle tubes are marked here by generic identifiers *m* and *n*.

P-Rules have a two-fold effect on design generation. When creating new structures using a generate-and-test method, there can be a high probability of being unable to apply a C-Rule in highly constrained situations, leading to a dead end and a restart of the algorithm. P-Rules can perturb the design to satisfy constraints that are blocking progress, such as moving existing parts that are obscuring connection points for new elements, so as to enable subsequent application of a C-Rule. Hence, if application of a C-Rule has failed, P-Rules may be applied for a number of iterations before C-Rule application is attempted again. This process is continued until an upper loop limit has been reached. If constraints are still not satisfied the algorithm restarts, otherwise, if C-Rule application has been successful, design generation can proceed.

P-rules also allow the sub-task of parametric optimisation of a structure representation for a given function graph to be studied using direct search. This search for preferred, often highly coupled, designs is a non-trivial problem, as free permutation of parametric design variables can violate inherent topologic (Table 1) and parametric (Table 2) constraints. Topologic constraints originate from relationships necessitated by functional requirements. For example, changes to the radius of gear disks need to maintain the correct ratio of the gear joint to satisfy the *mesh* constraint (Table 1). Remaining constraints, termed parametric constraints, uphold the sense of designs on a more basic level, ensuring that parts do not overlap, as well as safeguarding limits imposed on parametric values that are governed by problem-specific requirements, such as a maximum bounding box for a design or a minimum size for a power source element, and material limitations, such as a maximum gear disk size that can be structurally stable.

Table 1. Topologic Constraints.

Mesh	Ensure that gear disks interact in plane of spur gear, i.e. that sum of outer gear disk radii is equal to distance between spindles that gears disks are attached to: $\sqrt{([n].X-[m].X)^2+([n].Y-[m].Y)^2}$ $=[n(m)].OuterRadius+[m(n)].OuterRadius$ where *X*, *Y* are planar co-ordinates of spindles *[n]* and *[m]*, *[n(m)]* is gear disk on spindle *[n]* connecting to spindle *[m]* and vice versa.
Interact	Ensure that parametric elements interact in axial direction. For example for interacting gear disks: $[n(m)].Z2>[m(n)].Z2\ AND\ [n(m)].Z1<[m(n)].Z1\quad OR$ $[n(m)].Z2<[m(n)].Z2\ AND\ [n(m)].Z1>[m(n)].Z1$ where *[n(m)]* is gear disk on spindle *[n]* connecting to spindle *[m]* and vice versa. *Z1* and *Z2* are axial values of gear disk, i.e. height of gear disk is *Z1 – Z2*. This constraint is also relevant to interactions between gear disks and the spindles they are attached to.
Ratio	Ensure ratio of gear joints remain unchanged even if other parameters are changed. If gear joint ratio does change, ensure that overall ratio of gear train remains unchanged. $Let\ Ratio_n=\frac{[n(m)].OuterRadius}{[m(n)].OuterRadius}$ $Then\ \prod_{n=0}^{N}Ratio_n=RATIO$ where $n = 0, 1, 2, ..., N$ are all gear joints for which the total ratio must conform to a particular value *RATIO*, for example the gear joints between the hour and minute hands of a clock that require a ratio of 60.



1. set_spindle_top

Action: Adjusts position of top of spindle.

2. set_spindle_bottom

Action: Adjusts position of bottom of spindle.

Rationale: Allows spindles to be either lengthened to add more gear disks, or reduced in length to avoid taking up space that could be used more usefully by other elements. **Key constraint**: *interact*.

3. move_gears_vertically

Action: Moves a gear pair in the axial direction.

Rationale: Allows variation of gear positions, for example allowing torsion to be reduced if two sets of gear pairs are too far apart. **Key constraint**: *interact*.

4. move_spindle_angle

Action: Rotates a free spindle about its connecting spindle.

Rationale: Allows for rotation of free spindle to find a preferred valid location before connecting it to further spindles via new gear pairs. **Key constraint**: *mesh*.

5. set_spindle_r_out

Action: Sets outer radius value of a spindle.

6. set_spindle_r_in

Action: Sets inner radius value of a spindle.

Rationale: Allows the creation of a shell to provide concentric spindles. Can be used to create stiffer spindles or reduce mass of a spindle if a reduction of spindle stiffness is permitted.

7. set_gear_top

Action: Adjusts position of top of gear disk.

8. set_gear_bottom

Action: Adjusts position of bottom of gear disk.

Rationale: Changes contact area of gear disks which might be required by a particular gear mechanism. **Key constraint**: *interact*.

9. move_spindle

Action: Moves spindle in plane normal to axial direction.

Rationale: Allows rearrangement of spindles to create more favourable distribution of gear sizes, e.g. if a reduced aspect ratio of gear disks is required. **Key constraints**: *ratio*, *mesh*.

10. move_plate

Action: Moves a base plate in axial direction.

Rationale: Allows reduction in volume of overall clock. Allows more space to be created in clock structure if space is too crowded for further rule applications.

11. change_plate_radius

Action: Adjusts radii of base plates.

Rationale: Allows reduction in volume of overall clock by reducing the radius of base plates. Allows more space to be created within clock structure if space is too crowded.

Figure 4. P-Rules. Key topologic constraints (Table 1) are indicated if applicable.

P-Rules modify parameters of the structure representation while ensuring that all topologic and parametric constraints are upheld, so allowing parametric designs to be modified without changing the overall function of the design. Hence a like-for-like comparison can be used as a basis for search for preferred solutions if suitable metrics for judging the designs are available for use as objective functions.

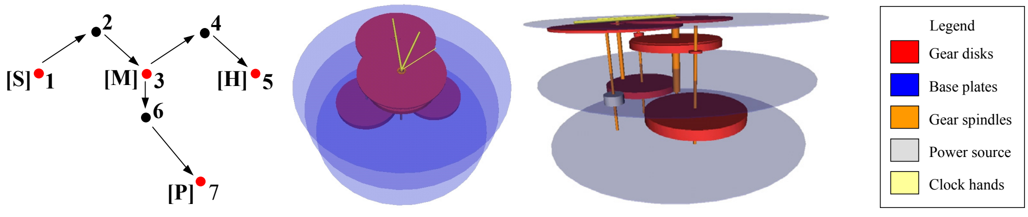
Table 2. Parametric Constraints

Collision	Ensure no collisions within the parametric structure, i.e. ensure no overlap.
Spatial	Ensure absolute and relative geometric parameters of elements fall within acceptable ranges, e.g. bounding box, minimum/maximum conditions: $Min\_Abs\_Param < [n].Param < Max\_Abs\_Param$ $Min\_Rl\_Param < [n].Param1 - [n].Param2 < Max\_Rl\_Param$ where *Param*, *Param1*, *Param2* are generic parameters, *Max_Abs_Param* and *Min_Abs_Param* are absolute spatial constraints, *Max_Rl_Param* and *Min_Rl_Param* are relative parameters.
Problem-specific	Equality constraints, usually problem-specific, for example the constraint on the spindles that carry the clock hands requiring them to be concentric: $[n].Param = [m].Param$, where *Param* is a generic parameter.
Component	Minimum / maximum component numbers *N*, e.g. $N_{\min} < N < N_{\max}$

4. Design synthesis results

4.1 Enhancing the generate-and-test algorithm

Following the above described algorithm, a successful design is created when the resulting structure is a valid representation of the target function graph. For the function graph shown in Figure 5a, a possible structural solution is shown in Figures 5b (top view) and 5c (side view). To demonstrate the application of P-Rules, the parallel grammar was used to create different designs for the function graph shown in Figure 5a, but using P-Rules to perturb interim designs as explained in the previous section when C-Rule application failed. Increasing the maximum number of P-Rules used raised the success rate of the algorithm.



(a) (b) (c)

Figure 5. (a) Function graph for design generation. Labels [S], [M], [H] and [P] refer to nodes that represent second, minute, hour hands and power source respectively. (b) and (c) show a parametric solution to (a).

Figure 6 shows different parametric designs generated with the parallel grammar algorithm. Each of these designs is one possible design from a whole family of possible parametric designs that fulfils the relevant function graph. Similarly, each function graph is a member of a whole group of function graphs with a similar pattern, but with a different numbers of nodes and connections, that make up a ‘clan’ of possible designs. Clans vary by basic connectivity between input and output in the function graph, families by the length of branches in the function graph and designs by the parametric values of the structural parts. Behaviour within a family varies with length of sub-branches, for example more nodes on a sub-branch indicates more gear joints, resulting in more potential for loss of mechanical energy through friction but allowing a reduction in ratio required at each gear joint and therefore less disparity between sizes of gear disks. Figure 7 shows how clans, families and designs fit into the overall design process. Moving from clans to families and from families to individual designs increases the level of design detail.

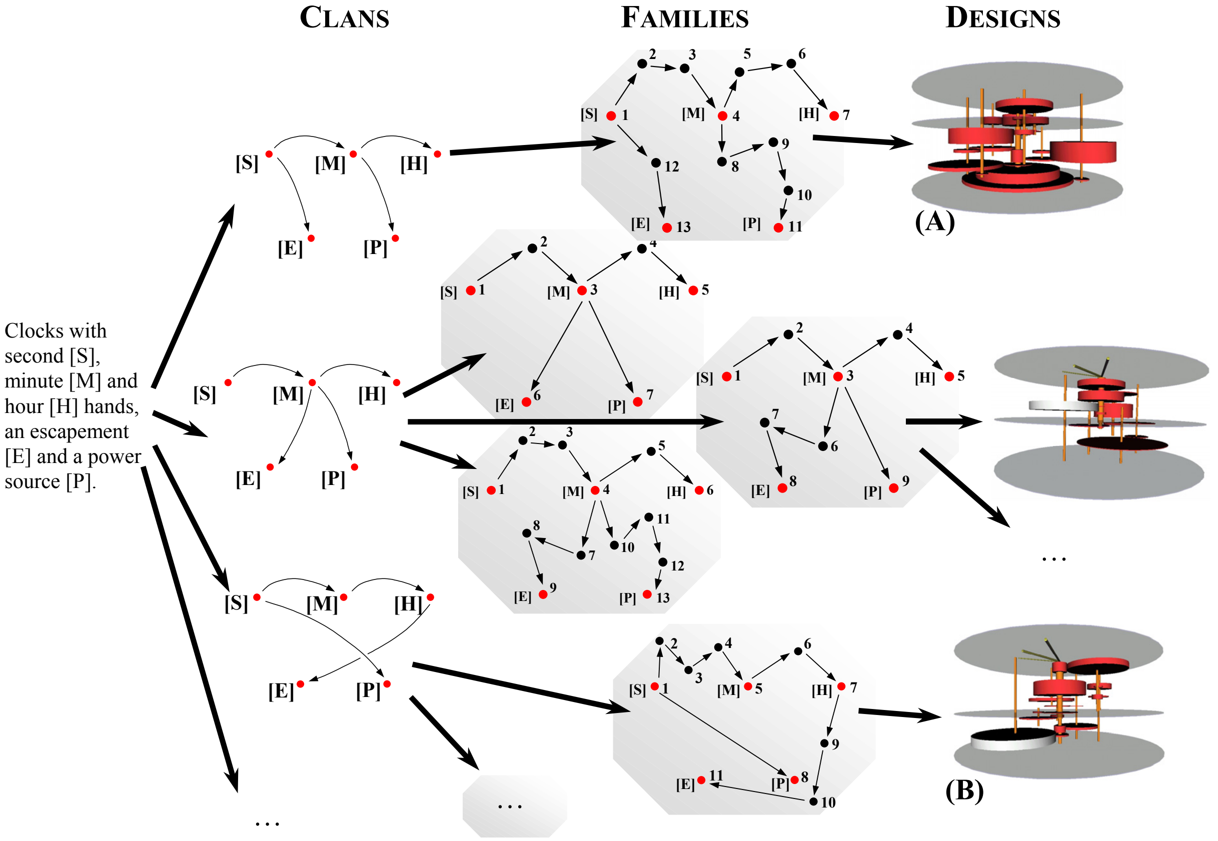


Figure 6. Clans and families of parametric designs. Label [E] indicates the clock escapement (timing mechanism). Designs A and B are used as examples in the next section.

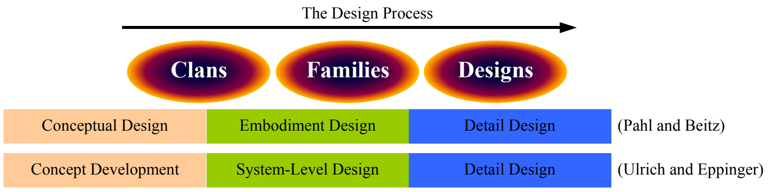


Figure 7. Clans, families and designs. Design process by Pahl and Beitz [18] and Ulrich and Eppinger [19]

4.2 Using direct search to generate preferred design solutions

The designs created in the previous section are initial design solutions that fulfil all the constraints on the system (Tables 1 and 2) to make them valid (Figure 2), but they are by no means optimal. We can find optimally directed designs by performing iterative search, evaluating designs using performance metrics to mediate among different designs. We consider the use of three metrics based on overall compactness, aggregate mass and total thickness, the minimisation (or, in some cases, maximisation) of which represent beneficial design characteristics of a clock or wristwatch.

Figure 8 shows preferred clock structures generated using mass and compactness metrics respectively. A mass reduction from 0.16 mass units (m.u.) to 0.06 m.u. was achieved for the former (Figure 8a), where gear disks were taken as brass and base plates and spindles as steel. A mass index is quite a simplistic metric to use, however, resulting in a preferred design that has only had some of the individual components reduced in size to a minimum given by spatial parametric constraints, for example a minimum sheet metal thickness for spur gears.

A compactness metric, calculated from the overall volume of the structure and the sum of the volumes of the individual elements, achieves a more closely packed solution (Figure 8b), achieving 10% height and 50% radius reductions in the example shown. This is a similar problem to the component packing problems addressed by Szykman and Cagan [16] with additional connectivity constraints introduced. The minimum radius achieved, 20.4 length units, is constrained by the minimum allowed size of the power source, given by the *spatial* constraint, and the size of its connecting gear disk, bound by the *ratio* constraint (Tables 1 and 2).

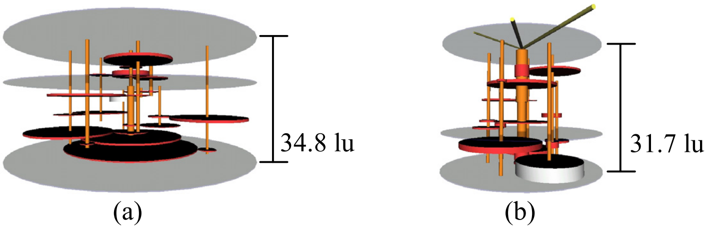


Figure 8. (a) Low mass design from initial design A in Figure 6. (b) Compact design from initial design B.

Figure 9 shows a preferred thin clock design, as might be designed for a wristwatch, for a similar function graph to the one shown in Figure 5a, i.e. with no escapement as the timing is controlled through the power source. Here reduction of overall thickness is the primary design goal, however, a secondary goal is required as some necessary design changes do not affect the thickness metric and so are not sought out by the generate-and-test method. For example, it is necessary to move gear pairs vertically using P-Rule 3 that are blocking the application of P-Rule 9 that moves base plates. Mass reduction is used as a secondary objective function as this shrinks the size of gear disks. Though this does not directly influence the primary objective function it is an important ingredient of the process required to find preferred designs as more space is created in the design, thus allowing the base plates to be moved closer together.

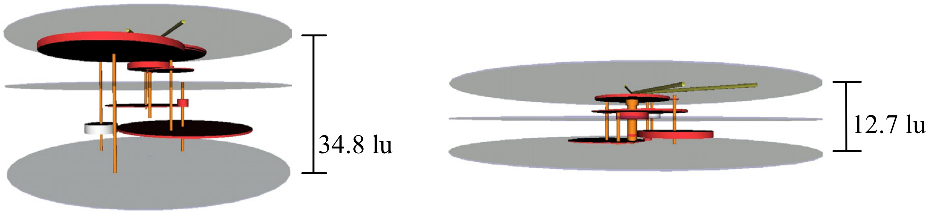


Figure 9. Initial design (left) and best thin design from a set of twenty experiments (right).

5. Discussion and further work

The new class of grammar rules, P-Rules, have made the parallel grammar more effective at generating valid structure representations of a given function graph. An alternate method of increasing the success rate of C-Rule applications that was considered would be to incorporate more knowledge into the C-Rule implementation itself, for example, allowing a C-Rule to calculate a ‘dimension for application’ to increase its chances of success. This avenue was not explored in detail as this would have increased the complexity of C-Rules and would have required increased domain-specific knowledge and, consequently, required higher maintenance when applied to new domains.

Work is planned to improve the motorised parts of a mechatronic camera design with a more sophisticated performance-based metric based on battery usage, calculated using dynamic simulation of virtual prototypes in the software package Modelica, to complement the existing spatially-based design metrics. This will show new designs being generated with the grammar and will allow the use of more sophisticated performance metrics.

There are several avenues of opportunity for further investigation. The parallel grammar has demonstrated how a grammar-based computational generation tool can enhance spatial innovation, however, contributions to functional innovation remain open ground for research. Progress in this area will address the issue of determining relevant design metrics for mechanical design, as spatially-based metrics do not fully capture the rationale behind designers’ decision-making. The introduction of P-Rules has also created a need for more sophisticated stochastic search processes, such as simulated annealing, to be utilised instead of the simple downhill search algorithm used to find preferred designs here. The search algorithm could potentially be improved by allowing computer analysis of the success rates of particular combinations of structure rules and then utilising this information to apply these combinations preferentially, i.e. using machine learning [20]. There is also potential to expand the grammar to investigate driving function rules from structure rules, generating new function graphs to produce further novel designs. The issue of general usability of grammars [21] must also be addressed, as the current implementation consists of a hard-coded grammar that requires developer-level interaction to alter grammar rules.

6. Conclusions

A mechanical design synthesis method based on a parallel grammar for design has been improved through the use of Perturbation Rules (P-Rules) to generate clans and families of parametric designs. The P-Rules have been used to improve the generation process that creates valid designs by increasing the success rate of the existing generate-and-test algorithm that is based on Create Rules (C-Rules). The P-Rules enable initial designs to be modified for comparison in the search for improved mechanical designs. As an example, preferred solutions with respect to low mass, thickness and compactness were generated for the mechanical clock example. Key extensions to this work include using the grammar for design in different domains, improving the search algorithms and enhancing performance metrics used.

7. Acknowledgements

Current research support is provided by the EPSRC (UK) and the Newton Trust (UK).

References

[1] Dym C.L., "Engineering Design: A Synthesis of Views", Cambridge University Press, 1994.

[2] Simon H.A., "The Structure of Ill-Structured Problems", Artificial Intelligence, Vol. 4, 1973, pp.181-201.

[3] Umeda Y., Tomiyama T., and Yoshikawa H., "Function, behaviour, and structure", Proceedings of the Applications of Artificial Intelligence in Engineering V: Design, Berlin, 1990, pp.195-211.

[4] Starling A.C. and Shea K., "A Clock Grammar: The Use of a Parallel Grammar in Performance-Based Mechanical Design Synthesis", Proceedings of the 2002 ASME International Design Engineering Technical Conferences, Montreal, Canada, 2002.

[5] Post E., "Formal reductions of the general combinatorial decision problems", American Journal of Mathematics, Vol. 65, 1943, pp.197-268.

[6] Antonsson E.K. and Cagan J., "Formal Engineering Design Synthesis", Cambridge University Press, Cambridge, 2001.

[7] Chakrabarti A., *et al.*, An approach to compositional synthesis of mechanical design concepts using computers, in Engineering Design Synthesis, Ed. A. Chakrabarti, 2002, Springer Verlag, London, pp.179-197.

[8] Liu Y.C., Chakrabarti A., and Bligh T.P., "Transforming Functional Solutions into Physical Solutions", Proceedings of the ASME Design Theory and Methodology Conference, Las Vegas, Nevada, 1999.

[9] Finger S. and Rinderle J.R., "A Transformational Approach to Mechanical Design using a Bond Graph Grammar", Proceedings of the First ASME Design Theory and Methodology Conference, Montreal, Quebec, 1989, pp.107-116.

[10] Paynter H.M., "Analysis and Design of Engineering Systems", MIT Press, Cambridge, USA, 1961.

[11] Lipson H. and Pollack J.B., "Evolution of Physical Machines: Towards Escape Velocity", Proceedings of the 6th International Conference on Artificial Intelligence in Design, AID'00, Worcester Polytechnic Institute, Worcester, Massachusetts, USA, 2000, pp.269-285.

[12] Schmidt L.C. and Cagan J., "Optimal Configuration Design: An Integrated Approach using Grammars", Journal of Mechanical Design, Vol. 120(9), 1998, pp.2-9.

[13] Li X., *et al.*, "Transformation of an EGT Grammar: New Grammar, New Designs", Proceedings of the ASME 2001 Design Engineering Technical Conferences, Pittsburgh, Pennsylvania, 2001.

[14] Chase S.C., "(Re)design of construction assemblies with function/behaviour/structure grammars", Proceedings of the Design e-ducation: Connecting the Real and the Virtual, Proceedings of the 20th Conference on Education in Computer Aided Architectural Design in Europe, Warsaw, 2002, pp.356-359.

[15] Bracewell R.H. and Johnson A.L., "From embodiment generation to virtual prototyping", Proceedings of the International Conference on Engineering Design, ICED 99, Munich, 1999.

[16] Szykman S. and Cagan J., "A Simulated Annealing-Based Approach to Three-Dimensional Component Packing", Journal of Mechanical Design, Vol. 117(June), 1995, pp.308-314.

[17] Shea K. and Starling A.C., "From Discrete Structures to Mechanical Systems: A Framework for Creating Performance-Based Parametric Synthesis Tools", Proceedings of the AAAI'03 Spring Symposium 'Computational Synthesis', Stanford, CA, 2003.

[18] Pahl G. and Beitz W., "Engineering Design", Springer, 1996.

[19] Ulrich K.T. and Eppinger S.D., "Product Design and Development", McGraw-Hill, 1995.

[20] Vale C.A.W. and Shea K., "Learning Intelligent Modification Strategies in Design Synthesis", Proceedings of the AAAI'03 Spring Symposium 'Computational Synthesis', Stanford, CA, 2003.

[21] Chase S.C., "A model for user interaction in grammar-based design systems", Automation in Construction, Vol. 11(2), 2002, pp.161-172.

Alex C Starling
Engineering Design Centre, Department of Engineering, Cambridge University, Trumpington Street
Cambridge CB2 1PZ, United Kingdom
Tel. +44 1223 766961, Fax. +44 1223 766963, acs29@cam.ac.uk