Evolution of Lace Knitting Stitch Patterns by Genetic Programming

Anikó Ekárt
Computer Science
School of Engineering and Applied Science
Aston University
B4 7ET Aston Triangle
Birmingham, United Kingdom
A.Ekart@aston.ac.uk

ABSTRACT

In this paper we study the generation of lace knitting stitch patterns by using genetic programming. We devise a genetic representation of knitting charts that accurately reflects their usage for hand knitting the pattern. We apply a basic evolutionary algorithm for generating the patterns, where the key of success is evaluation. We propose automatic evaluation of the patterns, without interaction with the user. We present some patterns generated by the method and then discuss further possibilities for bringing automatic evaluation closer to human evaluation.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search; J.6 [**Computer-Aided Engineering**]: [Computer-Aided Design]

General Terms

Algorithms, Design

Keywords

Genetic programming, creativity, representation, evaluation

1. INTRODUCTION

Knitting has been considered a simple pastime activity for women. It has not received much attention in the computing scientific community. There are three main accounts of knitting in the computing literature: Margaret Boden [2] compares computer programs to knitting patterns and argues for the striking similarity between the concepts of hand knitting and computer programming. Eckert and Stacey [3] study human creativity in knitwear design. As knitwear design is a closed domain with little computer involvement until now, it allows for insight into the ways humans are using their creativity when designing new artefacts. The third account is the visualisation of knitted fabric through the use of a physical particle system model for machine knitting by Meiner and Eberhardt [8].

Creativity as such is considered a capability that is specific to humans. There are many studies of human creativity available, for example in design [5, 9]. In the meantime, many researchers would question whether computers can ever become creative. Evolutionary design and art in particular are the areas that try to prove via examples that computer can be creative [1]. The human-competitive results competition organised at GECCO since 2004 can also be seen as a display of computer creativity.

The present study is another small step toward showing that computers can be creative.[1] We describe a methodology for automatically generating lace knitting stitch patterns through evolutionary computation. Although creating or choosing the pattern is only one step in the process of designing knitwear, its automation will lead closer to the understanding and automation of the whole knitwear design process.

2. KNITTING CHARTS

Knitting charts are used to represent knitting stitch patterns in an easy-to-understand, space saving graphical way. The frequently used symbols in lace knitting are shown in Fig. 1.[2] To each symbol there is associated a brief textual description explaining the knitting steps to be performed by the knitter wherever the symbol is met in the chart.

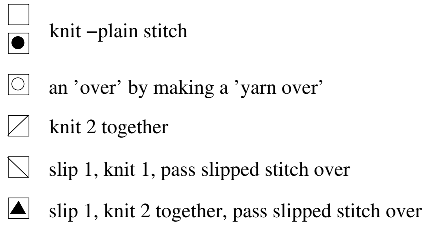

Figure 1: The symbols used in knitting charts

The knitting chart for an example pattern[3] is shown in Fig 2. The stitch pattern is knitted by reading the chart starting from the bottom right corner, toward the left, row by row. Only the odd rows are shown in the chart, as the even rows are very simple, i.e. they consist of repeating the same step: knitting purls.[4] The stitch pattern

GECCO'07, July 7–11, 2007, London, England, United Kingdom.

---

[1] We shall consider the definition of creative design based on the design product rather than the processes generating it.

[2] Note that pattern books published in different countries use different representations of the symbols.

[3] http://www.heirloom-knitting.co.uk

[4] Most lace knitting stitch patterns are of this kind.

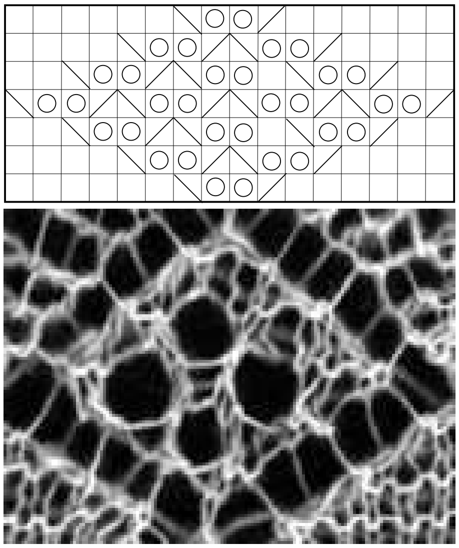

Figure 2: Lace hole diamond in diamond. Knitting chart and knitted sample

is repeated as many times as needed for the size of the garment, both in width and length. Each symbol on the knitting chart will have one resulting stitch in the corresponding row of the knitted pattern. The resulting stitches differ in appearance and in the fact that they are knitted on top of zero, one two or three stitches from the previous row. Each 'yarn over' *increases* the number of stitches by one (i.e. it is knitted on top of zero stitches from the previous row), whereas the final three symbols result in *decreasing* the number of stitches by one, one and two stitches, respectively (i.e. they are knitted on top of two, two and three stitches, respectively). As it can be seen, the 'yarn over's will result in holes in the lace pattern. If there are two 'yarn over's in a sequence, the hole will be larger than for one 'yarn over'. The / symbols will result in a stitch slanted to the right and the \ symbols in a stitch slanted to the left.

3. TREE REPRESENTATION OF KNITTING CHARTS

A very basic, straight-forward computer representation of knitting charts would transcribe the knitting chart into a matrix with one element for each stitch on the chart. However, this representation is not very natural for a pattern, especially when considering that a pattern must be knittable.

A knitting pattern represented by a chart can be seen as a collection of stitches in the final row, where each stitch has been produced by one of the following:

1. a plain stitch knit over a stitch in the previous row,
2. a 'yarn over',
3. knit two stitches from the previous row together,
4. slip one stitch, knit one stitch, pass slipped stitch over knitted stitch or
5. slip one stitch, knit two stitches together, pass slipped stitch over the knitted one.

Basically, each stitch in the final row is produced by following its description corresponding to its symbol on the knitting chart. Each stitch is knitted on top of zero (type 2: $\circ$), one (type 1: plain), two (types 3,4: /, \) or three stitches (type 5:$\triangle$) from the previous row. Similarly, each stitch in that row is produced on top of zero, one, two or three stitches from the row before, and so on. Therefore, we can decompose the pattern into parts, which correspond to the production of one stitch from the final row each, as shown in Fig. 3. Then each such part can easily be represented by a tree structure, as shown in Fig. 5. The knitting chart will have a corresponding ordered set of trees containing as many trees as there are stitches in the pattern (more exactly in the last row of the pattern).

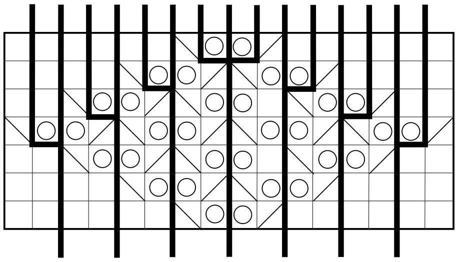

Figure 3: Decomposing the stitch pattern

For each chart there is a unique representation. On the other hand, any ordered set of trees can be produced to generate a knitting chart. More importantly, *each ordered set of trees represents a knittable pattern.*

By changing the order of the trees in the representation we could obtain different patterns.[5] For example, if trees 3 and 4 of Fig. 5 are swapped, the pattern shown in Fig. 4 results, where the modified part is highlighted.

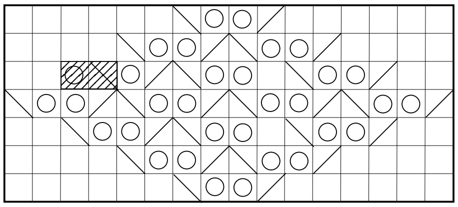

Figure 4: The pattern resulting from swapping trees 3 and 4 of the representation shown in Fig. 5

4. CONSTRAINTS

The representation described above guarantees that the pattern is knittable. More exactly the number of stitches left after each row is always equal to the number of stitches at the beginning of the next row. However, some knittable patterns may show an overall decrease or increase in the number of stitches, depending on whether the 'yarn over's or the three stitch number decreasing symbols dominate the pattern.

In order for the pattern to be feasible, the number of stitches in the final row must be the same as the number of stitches in the first row. The number of stitches in intermediate rows may vary slightly. This may result in knitting charts which are not exactly rectangular, with increases or decreases inside the pattern, but equal width at the start and the end.[6]

---

[5] A set of n trees can be used to represent $n!$ patterns by modifying the order of the trees.

[6] It could seem more natural to consider the stronger constraint of

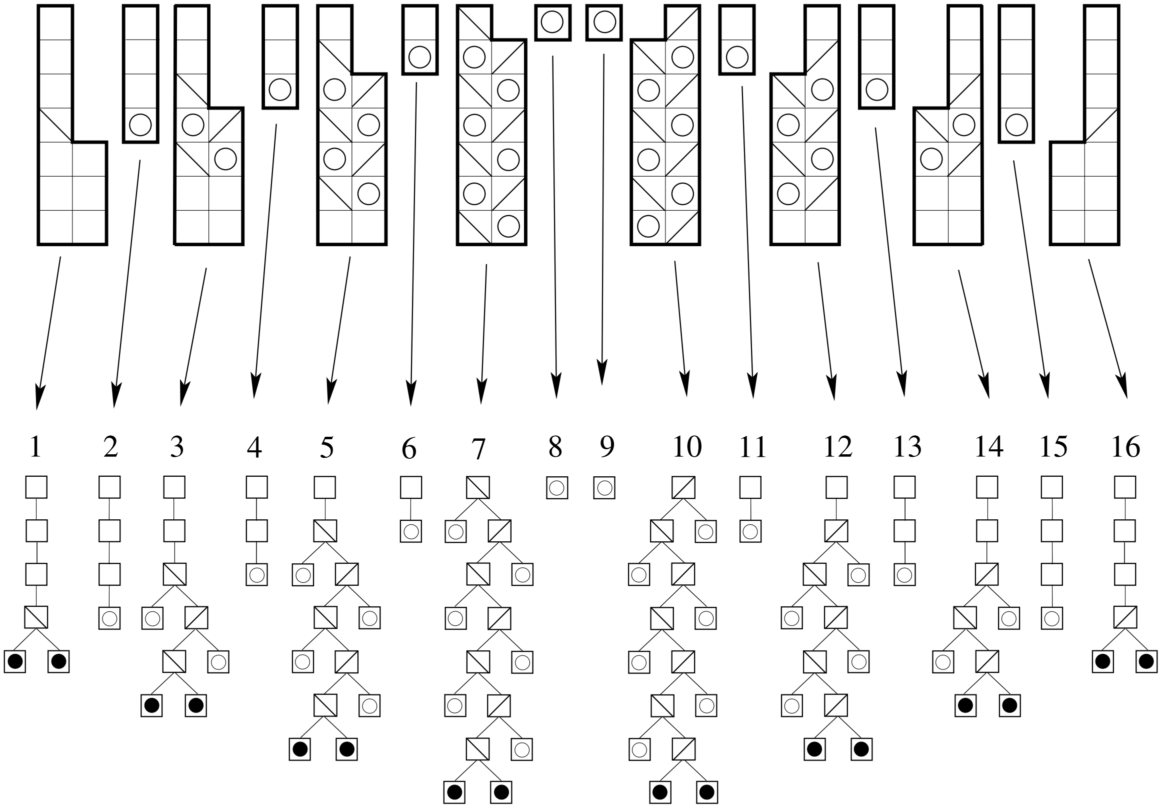

Figure 5: Genetic representation for "lace hole diamond in diamond" pattern

This constraint can be easily incorporated into the evolutionary pattern generation algorithm. As the proposed representation makes sure the pattern is knittable, it is sufficient to check that the overall number of increases is equal to the overall number of decreases for the whole pattern, i.e. set of trees:

$$|\circ| = |\backslash| + |/| + 2 \times |\triangle|, \qquad (1)$$

where $|\cdot|$ denotes the number of symbols $\cdot$ in the pattern.

The final constraint is that there cannot be more than two 'yarn over's next to each other simply because they could not be knitted and would produce too large, non-aesthetic holes in the lace.

5. THE EVOLUTIONARY ALGORITHM

Based on the tree representation described earlier, a tree based genetic programming system [6] can be used for evolving lace knitting charts for completely new patterns or as variations on existing patterns. The evolved population of genetic trees will represent a knitting pattern. The terminals correspond to plain stitches • and 'yarn over's ∘. The functions are: (1) plain stitches knit on top of one stitch from the previous row each, and(2) decreases $/, \backslash, \triangle$ knit on top of two, two and three stitches from the previous row. Each terminal and function produces one stitch, which can be the parameter for any function, i.e. the interpretation of a function parameter will be *a stitch on top of which the stitch corresponding to the function will be knit*.

The steps of the algorithm are the following:

1. Random size, shape and content trees are generated (if a new pattern from scratch is wanted) or an existing knitting chart is encoded (if variations starting from an existing pattern are desired).
2. The population is evaluated and if the resulted pattern is acceptable, the algorithm stops.
3. A generation of trees is produced by applying crossover and mutation to selected individuals in the population of trees.
4. The algorithm is continued from step 2.

The evaluation of the patterns represented by the genetic populations is automatically performed by the computer and is described in the next section.

6. AUTOMATIC EVALUATION

Most evolutionary art systems involve humans to some extent in the evaluation process mainly because it is very hard, if not impossible to define an appropriate evaluation function for aesthetic judgement. The difficulty in assessing each object on an absolute scale can be overcome by asking the evaluator for their preference among a number of objects instead. However, different people may consider different images as aesthetic, sometimes without being able to explain why they prefer one image over another. Several problems with human evaluation remain:

- user fatigue - the evaluator may become tired and cannot be expected to look at and judge too many images;
- inconsistency - the evaluator may not have the very same preferences over time;

---

having the same number of stitches in each row in order to have a rectangular chart, but as there exist patterns which are possible because of relaxing this constraint, we decided to implement the lighter version.

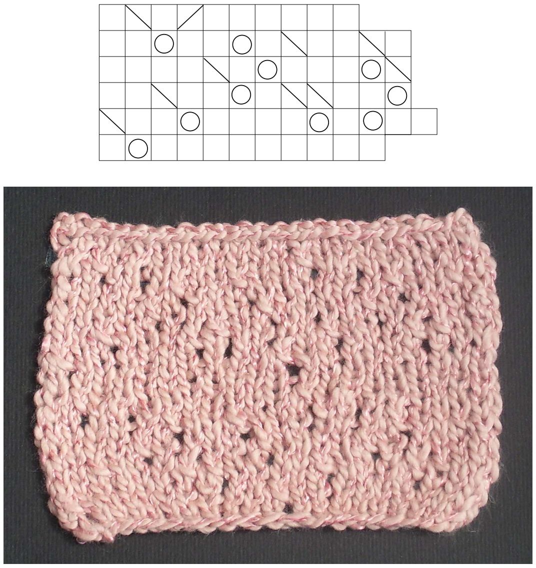

Figure 6: Pattern example 1. Knitting chart and knitted sample

- domain knowledge - the evaluator must have expertise in the particular field, in the present case knitting.

Therefore, it is highly desirable to simulate human judgement [7] and devise a method for automatic evaluation.

As the problem domain for lace knitting charts is constrained, the constraints have to be incorporated into the evaluation. We propose the use of multiple objectives for different aesthetic measurements in addition to constraint satisfaction. The aesthetic measurements were derived from personal experience and examination of a large number of lace knitting stitch patterns from pattern books.

Two main aesthetic measurements are proposed:

- $holes$ - the number of holes (○) to be maximised. This objective is based on the observation that lace patterns that contain many holes are nicer than the ones which contain few holes only.
- $knit\_length$ - the length of plain stitches knit on top of each other to be maximised. This objective is based on the observation that long lines of plain stitches in places in the pattern often result in very nice patterns.

Other aesthetic measurements, for example based on symmetry or connectedness of the holes can be implemented by including additional objectives. Also, if there is a preference for certain shapes in the pattern, new objectives for them can be developed and incorporated.

The objectives are not equally important: $holes$ is more important than $knit\_length$, therefore we use a model where the single objective of the weighted sum of the objectives is maximised:

$$\sum_{Maximise\ Obj_i} w\_i \times Obj\_i - \sum_{minimise\ Obj\_j} w\_j \times Obj\_j. \quad (2)$$

7. RESULTS

A prototype system has been implemented, which can produce knitting charts automatically. The simple genetic programming system proposed by Koza [6] is used with standard crossover and subtree mutation. Diversity maintenance through fitness sharing [4] ensures that the resulting patterns are not repetitions of smaller patterns.

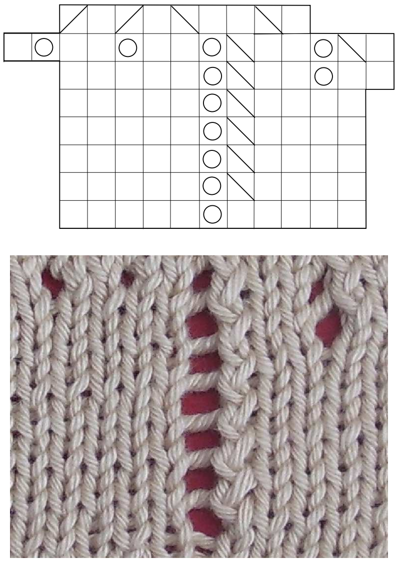

Figure 7: Pattern example 2. Knitting chart and corresponding knitted sample

We experimented with different evaluation functions ranging from simple constraint satisfaction to aesthetic measurements: number of plain stitches knit on top of each other $knit\_length$ and number of holes in a pattern $holes$.

An example created using constraint satisfaction only is shown in Fig. 6. If only looking at the chart shown in the top part of the figure, it is hard even for an experienced knitwear designer to envision the appearance of the knitted fabric. A knitted sample is shown in the bottom part of the figure, which reveals that this pattern is not particularly nice or special.[7]

Much better looking patterns can be obtained by using multi-objective evaluation including the $holes$ and $knit\_length$ objectives in addition to constraint satisfaction. An example is shown in Fig. 7. Although we can see the regularity in the pattern, when only looking at the knitted sample strictly corresponding to the chart, without repetitions, it is hard to appreciate the looks of the knitted fabric with the repeated patterns. The appearance of the knitted fabric can be fully appreciated from the larger knitted sample shown in Fig. 8.

The usual growing tendency of genetic trees, i.e. bloat, did not occur in any experiments. Actually, the trees tend to shrink rather than increase in size during evolution. This is due to the main constraint of having the same number of increases as decreases in the pattern described by Equation (1). The explanation is that it is easier to find shorter trees in the search space that satisfy the constraint. By incorporating the objective of $holes$ we ensure that the result is not the simple plain stitch pattern.

In order to encourage tree growth, actually translated into longer patterns, we experimented incorporating an objective for tree depth. One example pattern obtained is shown in Fig. 9. The appearance of the knitted fabric can be fully appreciated in Fig. 10. The increase in the number of stitches followed by the decrease is clearly noticeable on the sides of the fabric. A limitation on how much variation in size is allowed must be imposed so that the shape of the fabric is sufficiently close to a rectangle.[8]

---

[7] A human knitwear designer would probably call it *random*.

[8] This is necessary for the garment to look nice when worn.

Figure 8: Larger knitted sample for pattern example 2

8. CONCLUSION

A genetic programming method for automatically evolving lace knitting patterns with promising results has been presented. We provide evidence of artificial creativity through examples of aesthetic patterns evolved by a computer. The computer is not trying to imitate the very complex human creative processes, it is the product of artificial evolution that is being judged.

Automatic evaluation is achieved via multiple objectives. We envision introduction of more sophisticated objectives to produce *themed* patterns, such as waves, straight lines for both holes and decreases.

The introduction of specialised mutation operators (for example increasing or decreasing the length of plain stitches knit on top of each other or point mutation) is expected to improve the appearance of the patterns (for example a simple transformation of a \ into a / could lead to a better looking pattern, if the initial \ was in the middle of connected decreases of type /).

9. REFERENCES

[1] P. Bentley and D. Corne, editors. *Creative Evolutionary Systems*. Morgan Kaufmann, 2001.

[2] M. Boden. *Artificial Intelligence and Natural Man*. MIT Press, 1987.

[3] C. M. Eckert and M. K. Stacey. Adaptation of sources of inspiration in knitwear design. *Creativity Research Journal*, 2003.

[4] A. Ekárt and S. Németh. Maintaining the diversity of genetic programs. In J. Foster et al., editors, *Proceedings of the 5th European Genetic Programming Conference*, volume 2278 of *LNCS*, pages 162–171, 2002.

[5] J. S. Gero and M. L. Maher, editors. *Computational and Cognitive Models of Creative Design VI*. Key Centre of Design Computing and Cognition, University of Sydney, 2005.

[6] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. The MIT Press, Cambridge, Massachusetts, 1992.

[7] P. Machado, J. Romero, M. Ares, A. Cardoso, and B. Manaris. Adaptive critics for evolutionary artists. In *2nd European Workshop on Evolutionary Music and Art*, Coimbra, Portugal, April 2004.

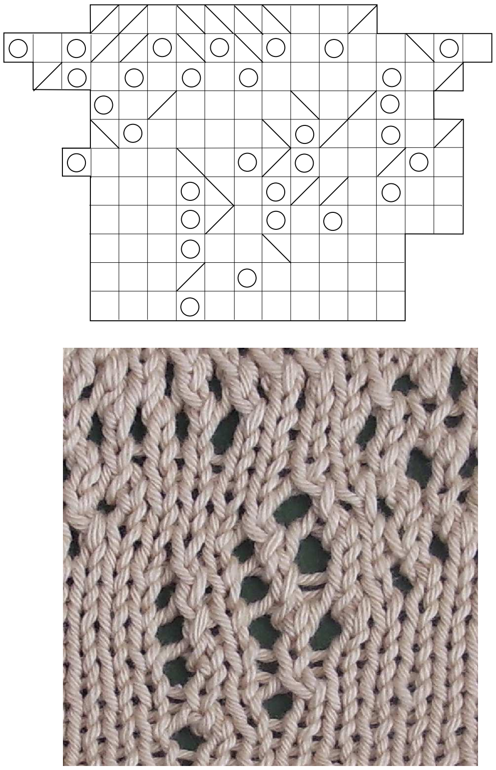

Figure 9: Pattern example 3. Knitting chart and corresponding knitted sample

Figure 10: Larger knitted sample for pattern example 3

[8] M. Meiner and B. Eberhardt. The art of knitted fabrics, realistic & physically based modelling of knitted patterns. In N. Ferreira and M. Gbel, editors, *EuroGraphics*, volume 17(3). Blackwell Publishers, 1998.

[9] R. Sosa and J. S. Gero. A computational study of creativity in design. *AIEDAM*, 19(4):229–244, 2005.