Automated Design of Part Feeders using a Genetic Algorithm*

Alan D. Christiansen **Andrea Dunham Edwards** **Carlos A. Coello Coello**

Department of Computer Science, Tulane University, New Orleans, LA 70118

Abstract

We describe a genetic algorithm approach to the automated design of vibratory bowl part feeders. Our approach gives us near-optimal designs in much less time than previously published optimal, brute-force search methods. We have implemented our approach in an automated part feeder design system, and we present preliminary results generated by our system.

1 Introduction

A recent report from the National Research Council [1993] has indicated a need for further research in the area of "rapidly reconfigurable production systems." The goal of such systems is to allow manufacturing lines to be quickly and easily modified to accommodate new products.

A prominent problem in manufacturing automation is the accurate and reliable presentation of small parts, in a desired orientation, to a workcell. This is often referred to as the "parts feeding" problem. Methodologies available for this problem include sophisticated computer vision-based bin-picking, manual loading of pallets, trays, or magazines, and the design of specialized feeding machines.

The design of automated feeder mechanisms (e.g., vibratory bowl feeders, centrifugal hoppers, magnetic feed hoppers) is currently more of an art than a science. However, recent work by Boothroyd [1992] and others has shown a methodology for designing such feeders, based on knowledge of part geometry and part material properties. The term "programmable feeder" has been used to describe a machine which can be adapted to new parts, rather than handling only one kind of part. Usually, this adaptation involves the redesign and replacement of the physical "tracks" on which parts move. The present paper presents a framework for the automated design of part feeder tracks.

*The research reported here was supported in part by Grants NSF/LEQSF (1992-93)-ADP-04 and NSF IRI-9410461.

Specialists who can design effective parts presentation systems are in great demand. With the current trend toward flexible assembly systems, smaller batch sizes, and the attendant assembly line reconfigurations, the problem of adapting feeder designs to new parts has become very important. Human expert designs for feeders are quite successful, but the prospect of *automation* of part feeder design and tuning has several rewards:

- potentially faster and cheaper setup during assembly line reconfiguration.
- codification and preservation of specialized expertise used by human specialists in feeder design.
- an expanded class of feeder designs through empirical modeling when mechanical analysis is difficult (e.g., curved part shapes, slightly deformable parts).
- automatic refinement of drive parameters and gate configurations to maximize throughput of the feeder.
- sensor-based on-line monitoring of feeder effectiveness, with automatic diagnosis of faults and suggestion of repairs.

In the present paper, we concentrate on the problem of determining the physical layout of tracks for vibratory bowl feeders.[1] In such devices, small parts travel from the bottom of a bowl along a track that spirals around the inside of the bowl. A vibratory motion causes the parts to climb up the track toward the lip of the bowl. Along the tracks are a sequence of "gates" which either (1) reorient parts as they pass along the track or (2) reject parts (i.e., cause parts to fall off the track back into the bottom of the bowl) if they are not in a desired orientation.

In general, the track layout design problem has two components:

[1] See [Goldberg *et al.* 1991] and [Goldberg *et al.* 1995] for related work using a different part feeding mechanism.

1. *Given* a part which we desire to feed, *determine* a sufficient set of gates for reorienting and rejecting the part.

2. *Given* a set of gates from which to choose, and a desired output orientation, *determine* an ordered sequence of gates along the track which will produce the highest feeder efficiency (highest fractional throughput of parts exiting in the desired orientation) and no parts exiting in undesired orientations.

We focus on problem 2 and use previously published results [Boothroyd 1992; Murch and Boothroyd 1971] for problem 1. Our approach, which we describe in the next two sections of the paper, uses a genetic algorithm. In the next section we discuss genetic algorithms in a general setting. Then we describe how we use the genetic algorithm framework for the problem of part feeder track design. Following the description of the method, we present four designs which were automatically derived by our system.

2 Genetic Algorithms

The *genetic algorithm* (GA) [Holland 1975] [Goldberg 1989] is a highly parallel search algorithm that is based on the mechanics of natural selection (survival of the fittest). Its basic operation is illustrated by the following segment of pseudo-code [Buckles and Petry 1992]:

```
generate initial population, G(0);
evaluate G(0);
t:=0;
repeat
   t:=t+1;
   generate G(t) using G(t-1);
   evaluate G(t);
until a solution is found
```

During the first generation, a set of individuals (each of which represents a potential solution to the problem we are dealing with) is generated randomly. The essential characteristics of each individual are encoded by a *chromosome*, normally of fixed length. We associate a *fitness function* with each individual, in order to determine how "good" the individual's solution is with respect to the rest of the population. This fitness function has to be provided by the user, and it should clearly reflect the importance of a certain solution over the others. We also need to choose the most suitable representation for encoding these solutions within the GA. The most common approach is to use a binary representation, although it is also possible to use letters, or a representation in which each *gene* (any position along the length of a chromosome) is a small integer [Michalewicz 1992].

By applying the corresponding fitness function, a *selection* procedure takes place, so that the fittest individuals reproduce, providing additions to the population of the next generation. There are two main selection schemes:

1. *Roulette wheel:* Under this approach, each individual is assigned a certain probability F_i of being selected, computed according to the formula $F_i = f_i/(\sum_j f_j)$, where f_i is the fitness value of each chromosome i [Buckles and Petry 1992]. Notice that even though the fittest individuals have a higher probability of being selected, individuals with lower fitness can also eventually be selected.

2. *Tournament:* In this approach, the population is shuffled and then is divided into groups of k elements from which the best individual (i.e., the fittest) will be chosen. This process has to be repeated k times because on each iteration only m parents are selected, where $m = (\textit{population size})/k$. For example, if we use binary tournament selection ($k = 2$), then we have to shuffle the population twice, since in each stage half of the parents required will be selected. The interesting property of this selection scheme is that we can guarantee multiple copies of the fittest individual among the parents of the next generation.

After selection, reproduction takes place. The operator called *crossover* produces an exchange of genetic material of a pair of individuals, so that a—presumably fittest—population is created. The two main ways of performing crossover are:

1. *Single-point crossover:* In this approach, a position of the chromosome is randomly selected as the crossover point, and the genes from that point to the end of the string are swapped between the two selected chromosomes.

2. *Two-point crossover:* In this approach, two positions are selected randomly along the chromosome length, and the genes between them are swapped between the two selected chromosomes.

Another important genetic operator is *mutation*. This operator randomly changes a gene of a chromosome. If we use a binary representation, a mutation changes a 0 to 1 and vice-versa. If we represent genes with integers, a mutation changes a gene to a random

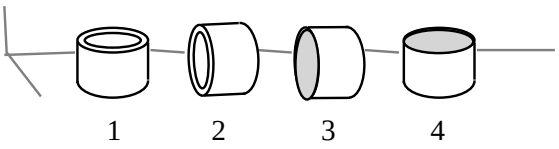

Figure 1: The four qualitatively distinct orientations of the cup part.

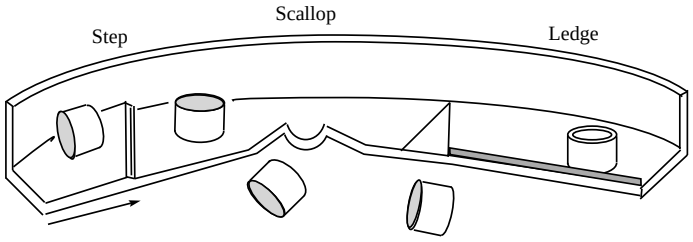

Figure 2: The track design presented in [Boothroyd 1992].

integer in the valid range of integers. This operator allows the introduction of new genetic material to the population and, from the theoretical perspective, it assures that—given any population—the entire search space is connected [Buckles and Petry 1992].

Since the GA is a heuristic technique, we cannot guarantee that the optimum solution will always be found, and we cannot even guarantee convergence. Because of this, we normally run several tests to fine tune the GA's parameters. To stop any run, two main criteria are used: either stop after a specified number of generations, or verify when the population has stabilized (i.e., all or most of the individuals have the same fitness).

3 Design of Gate Sequences

The design of a gate sequence can best be seen through an example (taken from [Boothroyd 1992]). Figure 1 illustrates four distinct orientations of a cup-shaped part, which we desire to feed. Orientation (1), with the open end of the cup up, is the desired output orientation for the feeder. Other possible orientations include the cup open end down (4) and the cup on its side with the open end either trailing with respect to the feed direction (2) or leading with respect to the feed direction (3). Boothroyd presents the example of the track design of Figure 2.

There are three gates in this design, in the sequence "step", "scallop", and then "ledge". The purpose of the step is to increase the proportion of parts in orientation (1). The purpose of the scallop is to eliminate (reject back in to the bottom of the bowl) parts in orientation (4). The purpose of the ledge is to reject parts in orientations (2) and (3). Thus, since (4) was eliminated by the preceding scallop, the net effect of the scallop-ledge sequence is to filter out all orientations except (1), the desired output.

Boothroyd models the effects of gates on parts by stochastic matrices. For example, for a step height of 7 mm on a cup part with a diameter of 12.7 mm and a height-to-diameter ratio of 1.132, the step gate is modeled by the matrix

$$\begin{bmatrix} 0.50 & 0 & 0.50 & 0 \\ 1.00 & 0 & 0 & 0 \\ 0.30 & 0 & 0.64 & 0.06 \\ 0.80 & 0.20 & 0 & 0 \end{bmatrix}$$

This matrix indicates that a part in orientation (1) has an equal likelihood (50% each) of exiting in orientation (1) or (3). This is seen from the top row of the matrix. Similarly, the second row says that a part in orientation (2) will exit in orientation (1). The third row indicates that a part in orientation (3) will exit in either (1) (30%) or (3) (64%) or (4) (6%). The last row indicates that a part in orientation (4) will exit either in orientation (1) (80%) or (2) (20%).

The net effect of a sequence of gates is naturally modeled by a matrix multiplication of the gate models. For the step-scallop-ledge sequence, the result is

$$\begin{bmatrix} 0.50 & 0 & 0.50 & 0 \\ 1.00 & 0 & 0 & 0 \\ 0.30 & 0 & 0.64 & 0.06 \\ 0.80 & 0.20 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1.00 & 0 & 0 & 0 \\ 0 & 0.50 & 0 & 0 \\ 0 & 0 & 0.50 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1.00 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0.50 & 0 & 0 & 0 \\ 1.00 & 0 & 0 & 0 \\ 0.30 & 0 & 0 & 0 \\ 0.80 & 0 & 0 & 0 \end{bmatrix}$$

The resulting matrix indicates that *all* parts exit in orientation (1). Note that the rows of this matrix do not all sum to 1 because some parts are rejected back into the bowl. If we knew the probability distribution of orientations of parts first encountering the gate sequence, we could estimate the fraction of such parts that will exit the sequence in the desired orientation by multiplying a row vector representing the initial distribution times the resultant matrix above. This fraction

is termed the *efficiency* e of the feeder. A high efficiency is desirable because the effective feed rate of parts exiting the device is e times f, where f is the nominal feed rate of parts climbing the feeder track.

Boothroyd gives an empirically derived estimate of the initial distribution of cup parts as [0.27 0.35 0.35 0.03]. Hence the step-scallop-ledge system matrix results in an efficiency of 61.4%.

The design problem of interest to us is how to choose the gate sequence, given a desired output orientation and a set of gate models which can be used (possibly multiple copies of the same gate) in the sequence. For a design to be acceptable, the sequence must allow *only* parts in the desired orientation to exit. A "good" design is one which further has a high efficiency. Boothroyd assumes that this design process is left to a human expert, but the primary result of the present paper is that this design problem can be automated.

3.1 Implementation as a Genetic Algorithm

The track layout design problem is basically a search over a large multidimensional space of possible designs. Each design corresponds to a sequence of gates, each modeled by a stochastic matrix. Some sequences of matrices produce valid designs (all parts exiting in the desired orientation) and some valid designs are better than others in terms of their efficiency. Thus the problem requires searching for near-optimal solutions in a large multidimensional space, which is the hallmark of a problem well-suited to a genetic algorithm approach.

Our desired solution is a sequence of matrices, which when multiplied together yields a result with all zeros in the columns other than the desired output column, and which when multiplied by the initial distribution gives a high efficiency. Thus, the individuals in the GA population encode a sequence of gate labels (integers), and the fitness of an individual is the efficiency computed by multiplying the initial distribution times the product of the matrices represented by the labels. Given the matrix models of the available gates, the design process is a straightforward application of the genetic algorithm framework.

Our program produces gate sequences based on the following inputs: (a) the matrices modeling the available gates, (b) the initial probability distribution of part orientations, (c) the desired output orientation, (d) some parameters for the genetic algorithm (which can be left to their default values given below), and (e) a length upper bound for the desired gate sequence. In the experiments reported below, we use binary tournament selection, two-point crossover, a population size of 100, a mutation rate of 1%, a crossover probability of 80%, and we stop the GA after 50 generations. All of these parameters were empirically determined to give acceptable results. Our system is implemented in the C programming language, and it is very fast. The designs reported in Section 4 below were produced by the program in less than five seconds each on a Sun SPARCstation 5.

3.2 Relationship to Manipulation Planning

Designing a sequence of gates along a feeder track is analogous to designing a sequence of manipulation actions for a robot to apply to an object. Both robot plans and feeder tracks apply a goal-oriented sequence of transformations to the position and orientation of the part. Thus much of the literature on motion planning under uncertainty is relevant to the problem discussed in the present paper.

Our methods are most closely related to manipulation planning methods that use explicit probabilistic models [Goldberg 1990; Christiansen and Goldberg 1995; Brost and Christiansen 1996], although other methods (e.g., [Lozano-Pérez *et al.* 1984]) are also appropriate for this problem. In [Christiansen and Goldberg 1995] an explicit, exhaustive search method (called Method I in that paper) was described. When applied to the current problem, Method I finds optimal sequences of gates, given a sequence length bound k, but its computational complexity is exponential in k, making the method practical only for small values of k. The new GA approach reported in the present paper does not guarantee optimal designs, but its complexity is directly proportional to k (because it must multiply sequences of k matrices in order to evaluate the fitness function).

4 Experimental Results

We have applied our program to several part feeding scenarios, which we report below.

4.1 Orienting a Cup-Shaped Part (I)

Our program, when given the three gate models discussed above, reproduced Boothroyd's design sequence. This is interesting, but not remarkable, as there are only $3^3 = 27$ distinct sequences of gates, assuming that we can use multiple copies of the gates.[2]

We investigated using a longer sequence of gates, but for this particular set of three gate choices, the optimal result is of length three.

[2] If we can only use one copy of each gate, then there are only six distinct sequences. We will assume from here on that we can use as many copies of a gate as we desire.

4.2 Orienting a Cup-Shaped Part (II)

Boothroyd [1992] gives data for steps of varying heights. For a step of height 3 mm and the same cup-shaped part, the matrix is

$$\begin{bmatrix} 1.00 & 0 & 0 & 0 \\ 0.85 & 0.15 & 0 & 0 \\ 0 & 0 & 1.0 & 0 \\ 0 & 0.6 & 0 & 0.4 \end{bmatrix}$$

Given a length bound of three, and substituting this step model for the previous 7 mm step, our program came up with the same sequence as before (step-scallop-ledge), but the reported efficiency of 56.75% was slightly worse than for the previous case. However, if we allow the program to use a longer sequence of gates, it chooses a sequence with many copies of the step, followed by the scallop and ledge. For a sequence length bound of 10, our program found a length 9 design with efficiency 64.96%, which is better than Boothroyd's 61.4%.

In this situation, there is a stochastic benefit to using multiple step gates in sequence, as there is a tendency to move orientations toward (1), which is the only orientation not rejected by the scallop-ledge sequence.

4.3 Orienting a Cup-Shaped Part (III)

We increased the set of gates that can be used in the design, including steps of every height from 1 to 7 mm in 1 mm increments along with the scallop (sc), ledge (l), and wiper (w). Also included were a slope (slp) (i.e., a gentler version of a step) and a hole (h) that passes through only orientations (1) and (4). Each of these 12 gates has a matrix representation, but because of space limitations in this paper, these matrices are not listed. Allowing length 10 designs gives us a search space of size 12^{10}, which is too large for a practical exhaustive search.

Our program finds the design s7-slp-s4-s3-s2-slp-h-sc-l-s3,[3] which has an efficiency of 74.5%. As in the previous example, multiple copies of steps are used to encourage the parts to enter orientation (1) before other orientations are filtered out by the scallop and ledge. This result is substantially better than the original 61.4% efficiency design reported in [Boothroyd 1992]. However, Boothroyd never claimed that his reported design was near optimal.

4.4 Orienting a Screw-Like Fastener

Our final example is taken from [Murch and Boothroyd 1971], where a design for a screw-like fastener was presented. Figure 3 shows the five qualitatively distinct part orientations: on-head (1), head-trailing (2), head-leading (3), perpendicular (4), and on-point (5).

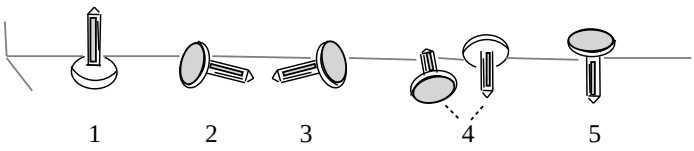

Figure 3: The five qualitatively distinct orientations of the screw part.

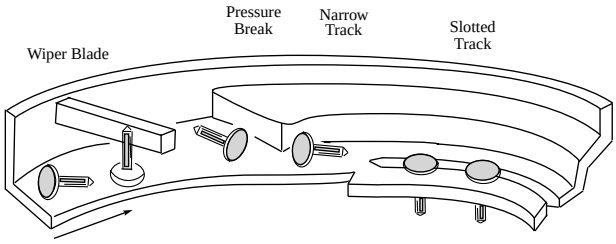

Figure 4: The track design presented in [Murch and Boothroyd 1971].

The four gate design reported in [Murch and Boothroyd 1971] is given by

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1.00 & 0 & 0 & 0 \\ 0 & 0 & 1.00 & 0 & 0 \\ 0 & 0 & 0 & 1.00 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1.00 & 0 & 0 & 0 & 0 \\ 0 & 1.00 & 0 & 0 & 0 \\ 0 & 0 & 1.00 & 0 & 0 \\ 0 & 0.25 & 0.25 & 0.50 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1.00 & 0 & 0 & 0 & 0 \\ 0 & 1.00 & 0 & 0 & 0 \\ 0 & 0 & 1.00 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1.00 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.00 \\ 0 & 0 & 0 & 0 & 1.00 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.00 \\ 0 & 0 & 0 & 0 & 1.00 \\ 0 & 0 & 0 & 0 & 0.50 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and is illustrated in Figure 4. It begins with a wiper to eliminate on-head orientations (and on-point orientations, although none are expected). It is followed by a pressure break which allows orientations (2) and (3) to pass while half of orientations (4) are passed through

[3] The notation sk means a step with height k mm.

and one quarter each are converted into orientations (2) and (3). Following this is a narrowed track rejecting (4). The final gate is a slotted track which turns the parts on-point (i.e., point-down) if they are in orientations (2) or (3). This on-point orientation is the desired output.

The initial probability distribution for the screw part was reported to be [0.40 0.20 0.20 0.20 0.00] in [Murch and Boothroyd 1971]. Hence this design has an efficiency of 50%.

When a length bound of four was used, our program found the following design: a wiper (w), followed by two pressure breaks (pb-pb), and finally ending with (the required) slotted track (st), for an efficiency of 55%. In this case the GA result indicates that the "narrow track" gate is better replaced by another copy of the pressure break. When a length bound of three was used, the GA found the sequence w-pb-st, with an efficiency of 50%, again indicating that the narrowed track is redundant.

Using a length bound of 10, our program found the design pb-pb-pb-pb-pb-w-pb-pb-pb-st, for an efficiency of 59.92%. This is almost 10% better than the design given in [Murch and Boothroyd 1971]. Note that using the wiper in the middle of the sequence has the same effect as using it at the beginning—the difference is solely aesthetic.

5 Summary

We have presented a new method, based on a genetic algorithm, for the automated gate layout of a vibratory bowl feeder. We have demonstrated how this method is applied to design problems for realistic parts. Our automated system finds designs which are at least as good as the previously published designs [Boothroyd 1992; Murch and Boothroyd 1971]. In cases where longer sequences of gates can be used, our system finds significantly better designs. Our approach is a practical and fast method for automating the gate layout design problem.

In the future we intend to augment our system to include on-line modeling of gates (i.e., automatically learning stochastic matrix representations through observation). We are also very interested in physically accurate dynamic simulations which could allow us to design appropriate gates from CAD models of parts. We believe that the automated design method reported here, when coupled with this future work, has the potential to make a marked change in the way vibratory part feeders are designed.

Acknowledgements

Discussions with Randy Brost and Matt Mason helped to crystallize the ideas presented here. We very much appreciate their comments and suggestions. Jim Jennings provided helpful comments on the paper's presentation.

References

[Boothroyd 1992] Boothroyd, Geoffrey 1992. *Assembly Automation and Product Design.* Marcel Dekker.

[Brost and Christiansen 1996] Brost, Randy C. and Christiansen, Alan D. 1996. Probabilistic analysis of manipulation tasks: A computational framework. *International Journal of Robotics Research* 15(1). in press.

[Buckles and Petry 1992] Buckles, Bill P. and Petry, Frederick E. 1992. *Genetic Algorithms.* Technology Series. IEEE Computer Society Press.

[Christiansen and Goldberg 1995] Christiansen, Alan D. and Goldberg, Kenneth Y. 1995. Comparing two algorithms for automatic planning by robots in stochastic environments. *Robotica.* In press.

[Goldberg *et al.* 1991] Goldberg, Ken; Mason, Matt; and Erdmann, Mike 1991. Generating stochastic plans for a programmable parts feeder. In *IEEE International Conference on Robotics and Automation.*

[Goldberg *et al.* 1995]
Goldberg, Ken; Craig, John; Zanutta, Rob; and Carlisle, Brian 1995. Estimating throughput for a flexible part feeder. In *Fourth Intl. Symp. on Experimental Robotics.*

[Goldberg 1989] Goldberg, David E. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning.* Reading, Mass. : Addison-Wesley Publishing Co.

[Goldberg 1990] Goldberg, Kenneth Yigael 1990. *Stochastic Plans for Robotic Manipulation.* Ph.D. Dissertation, Carnegie Mellon University, School of Computer Science.

[Holland 1975] Holland, John H. 1975. *Adaptation in Natural and Artificial Systems.* Ann Harbor : University of Michigan Press.

[Lozano-Pérez *et al.* 1984] Lozano-Pérez, Tomás; Mason, Matthew T.; and Taylor, Russell H. 1984. Automatic synthesis of fine-motion strategies for robots. *International Journal of Robotics Research* 3(1):3–24.

[Michalewicz 1992] Michalewicz, Zbigniew 1992. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer-Verlag, second edition.

[Murch and Boothroyd 1971] Murch, L. E. and Boothroyd, G. 1971. Predicting efficiency of parts orienting systems. *Automation* 18:55–57.

[National Research Council 1993] National Research Council, 1993. Information technology and manufacturing: A preliminary report on research needs. (Report from the Committee to Study Information Technology and Manufacturing).