A re-examination of the Cart Centering problem using the Chorus system

R. Muhammad Atif Azad
Dept. of Computer Science and Information Systems
University of Limerick
Ireland
atif.azad@ul.ie

Conor Ryan
Dept. of Computer Science and Information Systems
University of Limerick
Ireland
conor.ryan@ul.ie

Mark E. Burke
Dept. of Mathematics and Statistics
University of Limerick
Ireland
mark.burke@ul.ie

Ali R. Ansari
Dept. of Mathematics and Statistics
University of Limerick
Ireland
ali.ansari@ul.ie

Abstract

The cart centering problem is well known in the field of evolutionary algorithms and has often been used as a proof of concept problem for techniques such as Genetic Programming. This paper describes the application of a grammar based, position independent encoding scheme, *Chorus*, to the problem. It is shown that using the traditional experimental setup employed to solve the problem, Chorus is able to come up with the solutions which appear to beat the theoretically optimal solution, known and accepted for decades in the field of control theory. However, further investigation into the literature of the relevant area reveals that there is an inherent error in the standard E.C. experimental approach to this problem, leaving room for a multitude of solutions to outperform the apparent best. This argument is validated by the performance of Chorus, producing better solutions at a number of occasions.

1 Introduction

The cart centering problem is a well known problem that often appears in the introductory literature of optimal control (see [Athans, Falb, 66]). In its most basic form, it involves a cart of mass m moving in one dimension on a frictionless horizontal surface. The cart can be moving with any velocity v and can have any position x along the x-axis. The problem is to bring the cart to the origin in a position-velocity space with the values of both x and v approaching zero in minimum amount of time. The literature shows that the problem already has a well defined solution, which guarantees that the cart is centered in minimum amount of time. Genetic programming (GP) [Koza, 92] (pages 122 through 147) has been shown to have successfully solved this problem. The experimental setup described therein shows the absence of any success predicate, meaning that the system is free to wander in the solution space and come up with anything that minimizes the time required to center the cart.

This paper describes the application of a relatively new, position independent, evolutionary automatic programming system, Chorus [Ryan et al, 02a] on this problem. The system involves a genotype phenotype distinction and like [Horner, 96], [Paterson, 97], [Whigham, 95], and Grammatical Evolution (GE) [Ryan, Collins, O'Neill, 98] [O'Neill, Ryan, 01] evolves programs using grammars. While our aim initially for this paper was to demonstrate that Chorus could be successfully applied to the problem, we were surprised to discover that our results showed that the system produced expressions that were able to centre the cart in less time compared to the theoretical optimal control strategy. However a closer examination of the problem, as described in the control genre, reflects that the approach traditionally employed to solve the problem involves an inherent error. As a result there is no unique solution for this problem under the *circumstances.*

The paper first describes a context free grammar in Backus Naur form, which is used to partially specify the behaviour of Chorus, similar to the way in which one specifies **functions** and **terminals** in GP. We then describe the Chorus system and the process involving the mapping from a genotype to phenotype is discussed, with an example. Section 5 describes the application of Chorus on the cart centering problem, the theoretical background, the experimental setup and then discusses the results in the light of literature from control theory. Section 6 draws some conclusions based on the experiences and results presented in the paper.

2 Backus Naur Form

Backus Naur Form (BNF) is a notation for describing grammars. A grammar is represented by a tuple $\{N, T, P, S\}$, where T is a set of terminals, i.e. items that can appear in legal sentences of the grammar, and N is a set of non-terminals, which are interim items used in the generation of terminals. P is the set of production rules that map the non-terminals to the terminals, and S is a start symbol, from which all legal sentences may be generated.

Below is a sample grammar, which is similar to that used by Koza [Koza, 92] in his symbolic regression and integration problems. Although Koza did not employ grammars, the terminals in this grammar are similar to his function and terminal set.

```
S = <expr>

<expr>    ::=  <expr> <op> <expr>   (0)
             | ( <expr> <op> <expr>)(1)
             | <pre-op> ( <expr> )  (2)
             | <var>                (3)
<op>      ::= + (4) | - (5) | % (6)
             | * (7)
<pre-op>  ::= Sin (8) | Cos (9)
             | Exp (A)| Log (B)
<var>     ::= 1.0 (C) | X (D)
```

3 The Chorus System

Chorus[Ryan et al, 02a] is an automatic programming system based coarsely on the manner in which enzymes regulate the model of a cell. Chorus belongs to the same family of algorithms as *Grammatical Evolution* [Ryan, Collins, O'Neill, 98] [O'Neill, Ryan, 01], and shares several characteristics with it. In particular, the output of both systems is governed by a BNF grammar as above, and the genomes, variable length binary strings, interpreted as 8 bit integers (referred to as *codons*), are used to produce legal sentances from the grammar.

There is, however, a crucial difference. It concerns the interpretation of each codon, which, when being processed is **mod**ed with the *total* number of production rules in the grammar. Thus each codon represents a particular production rule, regardless of its position on the chromosome. This behaviour is different from GE, where an integer is **mod**ed with only the number of rules that are *relevant* at that point in time, and the meaning of a codon is determined by those that precede it, leading to the so-called "ripple effect"[Keijzer et al, 01].

For example, consider the individual:

18	28	32	27	42	17	18	31	27	14
45	46	45	18	27	55	65			

which can be looked upon as a collection of hard coded production rules. When moded with the number of rules in the grammar (see section 2), which in this case is 14, the same individual can now be represented as follows (using hexadecimal numbers):

4	0	4	D	0	3	4	3	D	0
3	4	3	4	D	D	9			

Each gene encodes a *protein* which, in our case is a production rule. Proteins in this case are enzymes that regulate the metabolism of the cell. These proteins can combine with other proteins (production rules in our case) to take particular *metabolic pathways*, which are, essentially, phenotypes. The more of a gene that is present in the genome, the greater the *concentration* of the corresponding protein will be during the mapping process [Zubay, 93] [Lewin, 99]. In a coarse model of this, we introduce the notion of a *concentration table.* The concentration table is simply a measure of the concentrations of each of the proteins at any given time, and is initialised with each concentration at zero. At any stage, the protein with the greatest concentration will be chosen, switching on the corresponding metabolic pathway, thus, the switching on of a metabolic pathway corresponds to the development of the forming solution with the application of a production rule.

Many decisions are made during the mapping process. For example, the start symbol `<expr>` has four possible mappings. When such a situation occurs, the relevant area from the concentration table is consulted and the rule with the maximum concentration is chosen. In case there is a tie, or the concentrations of all the rules are zero, the genotype is searched for any of the applicable rules, until a clear winner is found. This is analogous to the scenario where there are a number of voices striving for attention, and only the loudest is heard.

While searching for an applicable production rule, one may encounter rules that are not relevant at that point in time. In this case, the concentrations of those rules are increased, so when that production rule is involved in a decision, it will be more likely to win. This is what brings position independence into the system; the crucial thing is the presence or absence of a gene, while its position is less so. Importantly, *absolute* position almost never matters, while occasionally, *relative* position (to another gene) is important.

Once chosen, the concentration of that production rule is decremented. However, it is not possible for a concentration to fall below zero.

Sticking to the left most non-terminal in the current sentence, mapping continues until there are none left or we are involved in a choice for which there is no concentration either in the table or the genome. An incompletely mapped individual is given a fitness value of exactly zero in the current version of Chorus, thus removing its chances of indulging into any reproductive activity.

3.1 Example Individual

Using the grammar from section 2 we will now demonstrate the genotype-phenotype mapping of a Chorus individual. The particular individual is encoded by the following genome:

18	28	32	27	42	17	18	31	27	14
45	46	45	18	27	55	65			

For clarity, we also show the normalised values of each gene, that is, the genes mod 14. This is only done for readability, as in the Chorus system, the genome is only read on demand, and not decoded until needed.

4	0	4	D	0	3	4	3	D	0
3	4	3	4	D	D	9			

The first step in decoding the individual is the creation of the concentration table. There is one entry for each production rule (0..D), each of which is initially zero. The table is split across two lines to aid readability.

Rule #	0	1	2	3	4	5	6
Concentration							
Rule #	7	8	9	A	B	C	D
Concentration							

The sentence starts as `<expr>`, so the first choice must be made from productions 0..3, that is:

```
<expr>    ::=  <expr> <op> <expr>   (0)
             | ( <expr> <op> <expr>)(1)
             | <pre-op> ( <expr> )  (2)
             | <var>                (3)
```

None of these have a value yet, so we must read the first gene from the genome, which will cause it to produce its protein. This gene decodes to 4, which is not involved in the current choice. The concentration of 4 is incremented, and another gene read. The next gene is 0, and this is involved in the current choice. Its concentration is amended, and the choice made. As this is the only relevant rule with a positive concentration, it is chosen and its concentration is reduced, and the current expression becomes:

```
<expr><op><expr>
```

The process is repeated for the next leftmost non-terminal, which is another `expr`. In this case, again the concentrations are at their minimal level for the possible choices, so another gene is read and processed. This gene is 4, which is not involved in the current choice, so we move on and keep reading the genome till we find rule 0 which is a relevant rule. Meanwhile we increment the concentrations of rule 4 and D. Similar to the previous step, production rule #0 is is chosen, so the expression is now

```
<expr><op><expr><op><expr>
```

Reading the genome once more for the non-terminal `expr`, produces rule 3 so the expression becomes

```
<var><op><expr><op><expr>
```

The state of the concentration table at the moment is given below.

Rule #	0	1	2	3	4	5	6
Concentration					2		
Rule #	7	8	9	A	B	C	D
Concentration							1

The next choice is between rules #C and #D, however, as at least one of these already has a concentration, the system does not read any more genes from the chromosome, and instead uses the values present. As a result, rule `<var> -> X` is chosen to introduce first terminal symbol in the expression.

Once this non-terminal has been mapped to a terminal, we move to the next left most terminal, `<op>` and carry on from there. If, while reading the genome, we come to the end, and there is still a tie between 2 or more rules, the one that appears first in the concentration table is chosen. However if concentrations of all the relevant rules is zero, the mapping terminates and the individual responsible is given a suitably chastening fitness.

With this particular individual, mapping continues until the individual is completely mapped. The interim choices made by the system are in the order: $4, 3, D, 4, 0, 3, D, 4, 3, D$. The mapped individual is

```
X + X + X + X
```

The state of the concentration table at the end of the mapping is given in the next table.

Notice that there are still some concentrations left in the table. These are simply ignored in the mapping

Rule #	0	1	2	3	4	5	6
Concentration					2		
Rule #	7	8	9	A	B	C	D
Concentration							

process and, in the current version of Chorus, are not used again. Notice also that the rule #9 is not read because the mapping terminates before reading this codon.

4 Genetic Operators

The binary string representation of individuals effectively provides a separation of search and solution spaces. This permits us to use all the standard genetic operators at the string level. Crossover is implemented as a simple, one point affair, the only restriction being that it takes place at the codon boundaries. This is to permit the system to perform crossover on well-formed structures, which promotes the possibility of using schema analysis to examine the propagation of building blocks. Unrestricted crossover will not harm the system, merely make this kind of analysis more difficult.

Mutation is implemented in the normal fashion, with a rate of 0.01, with crossover occuring with a probability of 0.9. Steady state replacement is used, with roulette wheel selection.

As with GE, if an individual fails to map after a complete run through the genome, wrapping operator is used to reuse the genetic material. However, the exact implemenation of this operator has been kept different. Repeated reuse of the same genetic material *effectively* makes a wrapped individual behave like multiple copies of the same genetic material stacked on top of each other in layers. When such an individual is subjected to crossover, the stack is broken into two pieces. When linearized, the resultant of crossover is different from one or the other parent at regular intervals. In order to minimize such happenings, the use of wrapping has been limited to initial generation. After wrapping, the individual is flattened or *unrolled*, by putting all the layers of the stack together in a linear form. The unrolled individual then replaces the original individual in the population. This altered use of wrapping in combination with position flexibility, promises to maintain the exploitative effects of crossover. Unlike GE, the individuals that fail to map on the second and subsequent generations are not wrapped, and are simply considered infeasible individuals.

5 The Cart Centering Problem

The cart centering problem is well known in the area of evolutionary computation. Koza[Koza, 92] successfully applied GP to it, to show that GP was able to come up with a controller that would center the cart in the minimum amount of time possible.

The problem, also referred to as the double integrator problem, appears in introductory optimal control textbooks as the classic application of *Pontryagin*'s Principle (see for instance [Athans, Falb, 66]). There has been considerable research conducted into the theoretical background of the problem, and the theoretical best performance can be calculated, even though designing an expression to produce this performance remains a non-trivial activity.

As Evolutionary Computation methods are bottom up methods, they do not, as such, adhere to problem specific (be it theoretic or practical) information. This means that E.C. can be used as a testing ground for theories - if one can break the barriers proposed by theoreticians, then it probably means that there is a flaw in the theory concerned. However, another possibility is that there is a flaw in the experimental set up, that makes it appear as though the theoretical best has been surpassed.

This section describes the application of Chorus to the cart centering problem, an exercise which ppears to consistently produce individuals that surpass the theoretical best, before discussing the implications of the result.

5.1 Theoretical Background

In its most basic form, we consider a "cart" as a particle of mass m moving in one dimension with position at time t of $x(t)$ relative to the origin, and corresponding velocity $v(t)$. The cart is controlled by an amplitude constrained thrust force $u(t)$, $|u(t)| \leq 1$, and the control objective is to bring the cart to rest at the origin in minimum time on a frictionless track. The state equations are

$$\begin{aligned} \frac{dx}{dt} &= v \\ \frac{dv}{dt} &= \frac{1}{m}u \end{aligned}$$

or

$$\frac{d}{dt}\begin{bmatrix} x \\ v \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} u \qquad (1)$$

The solution is a unique "Bang-Bang " control ($u(t)$ takes only the values +1 or -1) with at most 1 switch which is expressible in feedback form ($u = u^*(x, v)$) in terms of a "switching curve" S in the $x - v$ plane. Following the approach of [Athans, Falb, 66] we find that S is given by

$$x + \frac{m}{2} v|v| = 0, \tag{2}$$

the optimal control by

$$u^* = \begin{cases} -1, & \text{if } x + \frac{m}{2}v|v| > 0 \\ +1, & \text{if } x + \frac{m}{2}v|v| < 0 \\ -v/|v|, & \text{if } x + \frac{m}{2}v|v| = 0 \end{cases} \tag{3}$$

and the minimum time T to reach $(0,0)$ from (x, v) by

$$T = \begin{cases} mv + \sqrt{2m^2v^2 + 4mx}, & \text{if } x + \frac{m}{2}v|v| > 0 \\ -mv + \sqrt{2m^2v^2 - 4mx}, & \text{if } x + \frac{m}{2}v|v| < 0 \\ m|v|, & \text{if } x + \frac{m}{2}v|v| = 0 \end{cases} \tag{4}$$

The above formulae assume that the system can switch precisely when condition (2) is met. In practice, this is only approximated. The engineering literature contains analyses of what happens when non-ideal switching (deadband and/or hysteresis) occurs using real hardware with the resultant cycling, chattering and steady state error. (see [Gibson, 63] for more details).

5.2 Experimental Setup

GP has been shown to be able to successfully evolve the time optimal control strategy (see [Koza, 92]). The same experimental setup is used by Chorus except where mentioned otherwise. The simulation essentially entails a discretisation of the problem so as to enable a numerical approximation of the derivatives involved. This is referred to as an Euler approximation of the differential equations given in (1), *i.e.,*

$$x(t+h) = x(t) + hv(t),$$

$$v(t+h) = v(t) + \frac{h}{m}u(t),$$

where m is the mass of the cart, h represents the time step size, $v(t+h)$ and $x(t+h)$ represent velocity and distance from the origin respectively at time $t+h$ and $v(t)$ and $x(t)$ represent velocity and distance from the origin respectively at time t. The desired control strategy should satisfy the following conditions.

> It should specify the direction of the force to be applied for any given values of $x(t)$ and $v(t)$.
>
> The cart approximately comes to rest at the origin, *i.e.,* the Euclidean (x, v) distance from the origin is less than a certain threshold.
>
> The time required is minimal.

The exact time optimal solution is characterised by the switching condition

$$-x(t) > \frac{v^2(t) Sign\ v(t)}{2|u_{\max}|/m}, \tag{5}$$

which applies the force in the positive x direction if the above condition is met and in the negative direction otherwise. Note that $u_{\max}$ represents the maximum value of $u(t)$, which is 1 here. The *Sign* function returns +1 for a positive argument and -1 otherwise. For the sake of simplicity m is considered to be equal to 2.0 kilograms and the magnitude of the force $u(t)$ is 1.0 Newtons, so that the denominator equals 1.0 and can be ignored. The experimental settings employed by Koza are summarised in table 1. Note that (5) does not incorporate the equality condition mentioned in (3).

Table 1: A Koza-style Tableau For The Cart Centering Problem.

Objective:	Find a time optimal bang-bang control strategy to center a cart on a one dimensional frictionless track.
Terminal Set:	The state variables of the system: x (position of the cart along X axis), v (velocity V of the cart) and -1.0.
Function Set:	+,-,*,%,ABS,GT.
Fitness cases:	20 initial condition points (x, v) for position and velocity chosen randomly from the square in position-velocity space having opposite corners, $(-0.75, 0.75)$ and $(0.75, -0.75)$.
Fitness:	Reciprocal of sum of the time, over 20 fitness cases, taken to center the cart. When a fitness case times out, the contribution to the sum is 10.0 seconds.
Hits:	Number of fitness cases that did not time out.
Wrapper:	Converts any positive value returned by an expression to +1 and converts all other values (negative or zero) to -1.
Parameters:	M = 500, G = 75
Success Predicate:	None.

The grammar used for the problem is:

```
S = <expr>

<expr>    ::=  <expr> <op> <expr>
             | ( <expr> <op> <expr>)
```

```
                | <pre-op> ( <expr> )
                | <var>
<op>        ::= + | - | % | * | GT
<pre-op>    ::= ABS
<var>       ::= X | V | -1.0
```

The randomly generated 20 fitness cases used by Chorus are given in the table 2.

Table 2: Randomly Generated 20 Starting Points, given as ((x, v) pairs).

0.50,0.67	-0.65,0.40	-0.16,-0.57	0.10,0.50
-0.71,0.66	0.43,0.01	-0.28,-0.71	0.27,-0.73
-0.50,0.34	-0.57,0.32	0.43,-0.69	-0.52,-0.16
-0.33,-0.21	-0.16,-0.06	0.71,-0.69	-0.04,-0.63
0.39,0.70	-0.52,-0.42	-0.59,0.38	0.58,-0.35

The cart is considered to be centered if the Euclidean distance from the origin $(0, 0)$ is less than or equal to 0.01. The total time taken by the strategy (5) over all the given set of starting points is 56.07996 seconds. On average it takes 2.803998 seconds per fitness case for the cart to be centered. This means that any strategy which centers the cart in less time, does better than the theoretical solution (5) for this experimental setup.

5.3 Experimental Results

The work of Koza [Koza, 92] shows that the optimal control strategy can be evolved using GP. However, it has not been shown that even in the absence of any success predicate, any strategy was evolved which could beat the result as described by the inequality (5). When the same task is given to the Chorus system, 17 times out of 20 independent runs, it evolves what appears to be a better strategy in terms of time minimisation. Out of those 17 runs, on the average, a better strategy is produced in the 39th generation, the earliest being 20th and the latest being 65th.

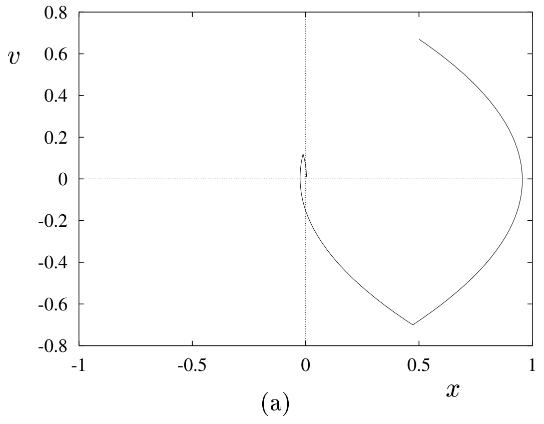
One of the samples which broke the barrier is given as

$$(-1.0 * X)\ GT\ (V * ABS(V) + V * V * V),$$

which can be rewritten as

$$-x(t) > v^2(t) Sign\ v(t) + v^3(t), \qquad (6)$$

returning $+1$ if the condition is satisfied and -1 otherwise. Total time recorded for this control law mentioned by inequality (6) is 50.799965 seconds over 20 fitness cases which is clearly less than the solution shown by the inequality (5). However, the least time that was recorded was 49.919968 seconds. A plot of x versus v for the control strategy given in (5) is shown in Fig 1(a) for the starting point $(0.50, 0.67)$. A similar plot for the strategy evolved by the Chorus system is shown in Fig 1(b). Notice that in (a) the control strategy crosses the y-axis leading into the negative x-axis region and then it returns to the origin. This shows the longer route traversed by (a) compared to (b) where there is no such occurrence, thus reflecting the time difference between the two strategies.



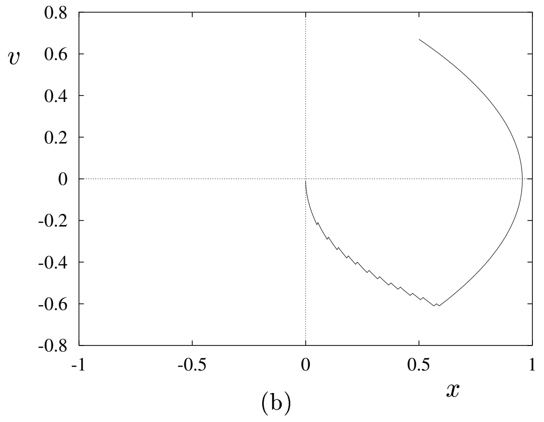


Figure 1: .Trajectories traversed by the two strategies to reach the origin. (a) represents inequality (5) and (b) represents the evolved strategy (6)

5.4 Discussion

It appears from the results in the previous section that a solution better than the theoretical has been achieved. However, a careful consideration of the problem undertaken shows otherwise. This problem has been solved by first discretising the main differential equations as mentioned earlier. The discretisation brings with it an element of error. The time step h

used now plays a major role, in the sense that a smaller time step would lead to a better solution *i.e.,* closer to the theoretical solution (3), and as $h \to 0$ the solution converges to (3).

The time step employed by Koza [Koza, 92] is $h = 0.02$, and using this time step, the error in the derivatives is substantial enough to cause the systems to converge to control laws other than the theoretical result in (3). In this sense Chorus actually validates this by evolving to what is a better solution than (5).

A study of the appropriate literature in the control theory genre indicates that the theoretical model is just that, theoretical. Practical implementation of a control system which brings the cart to the target position is not even "bang-bang" (i.e $u(t)$ is either +1 or -1). Instead, the magnitude of the applied force is any real number between 0 and 1.

One approach is to model the situation as one in which the control can change only at discrete-time steps, either as a sampled data system or a discretised version of eq(1). The former leads to state equations

$$\begin{aligned} x(t+h) &= x(t) + \delta v(t) + \frac{\delta^2}{2m} u(t) \\ v(t+h) &= v(t) + \frac{\delta}{m} u(t) \end{aligned}$$

where $1/\delta$ is the sampling rate. The latter, using an *Euler* discretisation scheme, leads to state equations

$$\begin{aligned} x(t+h) &= x(t) + hv(t) \\ v(t+h) &= v(t) + \frac{h}{m} u(t) \end{aligned}$$

where h is the step size.
When $\delta = h$, both models are of the form

$$\begin{bmatrix} x(t+h) \\ v(t+h) \end{bmatrix} = \begin{bmatrix} 1 & h \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x(t) \\ v(t) \end{bmatrix} + \begin{bmatrix} b \\ h/m \end{bmatrix} u(t) \tag{7}$$

where $b = h^2/2m$ for the sampled data model and $b = 0$ for the discretised model.

The control objective is again to bring the state of eq (7) to the origin in minimum time using a sequence of amplitude constrained controls $|u| \leq 1$. However, due to the discrete time steps, the solution of the problem is fundamentally different to that of the continuous time problem of (1). The optimal control is in general no longer unique, nor except for a set of isolated points in the $x - v$ plane is it Bang-Bang throughout. Hence there are different approaches and algorithms.

The more general problem in n dimensions was initially formulated in [Kalman, 57], and then analysed comprehensively in [Desoer, Wing, 61a] - [Desoer, Wing, 61c]. This analysis, when applied to the cart centering problem recursively constructs a sequence of convex sets $\{C_k\}$, where C_k is the set of states for which there exists an admissible input sequence which transfers the state to the origin in k time steps but no fewer ($C_0 = \{(0,0)\}$). For instance, if we want to centre the cart in 1 time step then C_1 represents the region of interest. For any $(x_1, v_1) \in C_1$, the cart is guaranteed to be centered in exactly 1 time step. In addition, a piecewise linear switching curve is constructed which divides the plane into regions of positive and negative control values (see figure 2).

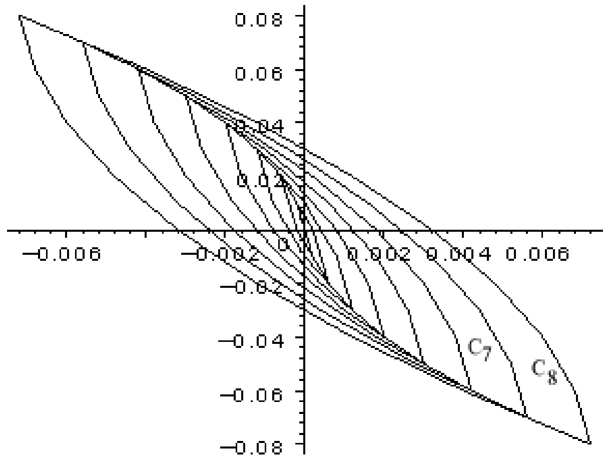


Figure 2: The sets C_1 - C_8 for the *Euler* discretised system with a cart of mass $m = 2$ and $h = 0.02$

Later work has looked at describing the C_k in terms of their facets with associated algorithms [Keerthi, Gilbert, 87], and there is still much interest in improving the efficiency of the existing algorithms (see [Jamak, 00] for a good review).

6 Conclusions

We have described the application of a position independent, representation scheme for Evolutionary Algorithms, termed Chorus, on the cart centering problem. Much to our surprise, Chorus apparently succeeded in producing individuals that performed better than the theoretical best. However, further analysis of the problem and traditional experimental set up revealed flaws that changed the nature of the problem.

The paper describes how Chorus was able to exploit these flaws to produce surprisingly fit individuals, and how an Evolutionary Computation system can be used to help test models of physical systems. Also, it reemphasizes the point that while attempting to solve continuous problems numerically, we should be ware of the resultant discretisation errors. It is also worth noting that the way these problems are typically solved by control engineers is by starting with the discretised analogues of the continuous problems and then proceeding to solve. It might be worth exploring what a system like Chorus may have to offer in the solution process of such a discretised problem.

6.1 Future Work

The results shown in the cart centering problem encourage the use of Chorus for real world problems. Coupled with the strengths of the system discussed in [Ryan et al, 02a], the system can be applied but not limited to the problems in the field of control theory and fluid dynamics.

Chorus diverges considerably from algorithms in the same "family", e.g. Grammatical Evolution and GAUGE[Ryan et al, 02b] in that it does not exploit the ripple effect, and instead uses position independent, absolute genes. This makes Chorus very suitable for schema analysis, and also possible that the Genetic Algorithm Schema Theory could, with very little extension, be applied to an automatic programming system.

References

[Athans, Falb, 66] Athans M. and P. L. Falb, *Optimal Control*, McGraw-Hill, 1966.

[Desoer, Wing, 61a] Desoer C.A. and J. Wing, "An Optimal Strategy for a Saturating Sampled-Data System", *IRE Transactions on Automatic Control*, vol AC-6, pp. 5-15, 1961.

[Desoer, Wing, 61b] Desoer C.A. and J. Wing, "A Minimal Time Discrete System", *IRE Transactions on Automatic Control*, vol AC-6, pp. 111-125, 1961.

[Desoer, Wing, 61c] Desoer C.A. and J. Wing, "Minimal Time Regulator Problem for Linear Sampled-Data Systems (General Theory)", *J. Franklin Inst.*, vol 20, pp. 208-228, 1961.

[Gibson, 63] Gibson J.E., *Nonlinear Automatic Control*, McGraw-Hill, 1963.

[Goldberg, Korb, Deb, 89] Goldberg D E, Korb B, Deb K. Messy genetic algorithms: motivation, analysis, and first results. *Complex Syst. 3*

[Horner, 96] H. Horner, *A C++ class library for Genetic Programming: The Vienna University of Economics Genetic Programming Kernel.* Release 1.0, Operating instruction. Vienna University of Economics, 1996.

[Jamak, 00] Jamak A., "Stabilization of Discrete-Time Systems with Bounded Control Input", MASc Dissertation, University of Waterloo, Waterloo CA.,2000.

[Kalman, 57] Kalman, R.E., "Optimal Nonlinear Control of Saturating Systems by Intermittent Action" , 1957 IRE Wescon Convention Record, pt 4, pp 130-135.

[Keerthi, Gilbert, 87] Keerthi S.S. and E.G. Gilbert, "Computation of Minimum-Time Feedback Control Laws for Discrete-Time Systems with State-Control Constraints", *IEEE Transactions on Automatic Control*, vol AC-32, pp. 432-435, 1987.

[Keijzer et al, 01] Keijzer M., Ryan C., O'Neill M., Cattolico M., Babovic V. Ripple Crossover In Genetic Programming, in *Proceedings of EuroGP'2001.*

[Koza, 92] J. Koza. "Genetic Programming". MIT Press, 1992.

[Lewin, 99] Lewin B. *Genes VII.* Oxford University Press, 1999.

[O'Neill, Ryan, 01] O'Neill M., Ryan C. Grammatical Evolution. *IEEE Transactions on Evolutionary Computation.* 2001.

[Paterson, 97] N. Paterson and M. Livesey, "Evolving caching algorithms in C by GP" in *Genetic Programming 1997: Proc. 2nd Annu. Conf.*, MIT Press, 1997, pp. 262-267. MIT Press.

[Ryan, Collins, O'Neill, 98] C. Ryan, J.J. Collins and M. O'Neill, "Grammatical Evolution: Evolving Programs for an Arbitrary Language", in *EuroGP'98: Proc. of the First European Workshop on Genetic Programming* (Lecture Notes in Computer Science 1391), Paris, France: Springer 1998, pp. 83-95.

[Ryan et al, 02a] C. Ryan, A. Azad, A. Sheahan and M. O'Neill, "No Coercion and No Prohibition, A Position Independent Encoding Scheme for Evolutionary Algorithms - The Chorus System". In the *Proceedings of European Conference on Genetic Programming (EuroGP 2002).*

[Ryan et al, 02b] C. Ryan, Miguel Nicolau, and M. O'Neill, "Genetic Algorithms Using Grammatical Evolution". In the *Proceedings of European Conference on Genetic Programming (EuroGP 2002).*

[Whigham, 95] P. Whigham, "Grammatically-based Genetic Programming" in *Proceedings of the Workshop on GP: From Theory to Real-World Applications*, Morgan Kaufmann, 1995, pp. 33-41.

[Zubay, 93] Zubay G. Biochemistry. Wm. C. Brown Publishers, 1993