Revision Lecture: Nature Inspired Design

Module Summary: What have we covered?

1. Engineering design and design optimisation
2. Creative design
3. Evolutionary art (graphics, music, ...)
4. Development rules
5. Circuit design and evolvable hardware
6. Molecular design and drug discovery

Examples: The emphasis is on understanding and problem solving.

Engineering Design and Design Optimisation

Engineering design problems are often formulated as an optimisation problem. For example,

1. designing a shortest route among N locations on a map can be formulated as TSPs,
2. designing an ANN with the best generalisation can be formulated as minimising an error function,
3. designing a network of pipes involves cutting long stocks into shorter pieces while minimising the wastage — the cutting stock problem,
4. ... ...

However, design is not always the same as optimisation.

Creative Design

1. In contrast to optimisation, creative design is far more explorative than exploitative. Evolutionary algorithms, if used, are primarily used as discovery engines rather than optimisation tools.
2. How creative an algorithm might be depends heavily on the representation and operators used.
3. Representation can be evolved too!

Engineering Optimisation vs. Creative Design

Engineering Optimisation: Parameterised representation, well-defined fitness function and optimality, well-defined and fixed search space, optimality is essential, exploitive in nature.

Creative Design: Flexible and adaptive representation, adaptive fitness function (or even without an explicit fitness function), search space not fixed, discovery is more important than optimality, explorative in nature.

Integrated Design: Creative design as the outer loop and engineering optimisation as the inner loop, e.g., in designing topology and geometry of a structure (truss).

Mapping Between Genotypes and Phenotypes

The issue of genotype-phenotype mapping occurs frequently in evolutionary design and is extremely important, especially when the mapping is not one-to-one.

- Fitness evaluation should be done on genotypes, but often done on phenotypes in practice. Potential danger ...
- Fitness sharing can be done on either genotypes or phenotypes. They can lead to very different results.
- The mapping from one space to another can made a problem simpler, e.g., smaller search space, smoother search space, constraint satisfaction.
- The mapping enables the use of a developmental process, which in turn enables the evolution of complex phenotypes from simple genotypes.

Design is Inherently Multi-Objective

- How to compare two solutions (fitness evaluation)?
- Finding a set, not a single solution — diversity is important
- Don't lose non-dominated solutions so far.

Evolutionary Art and Developmental Rules

Creative design can be achieved by both evolutionary and non-evolutionary approaches.

1. evolutionary graphics, images and music
2. Non-evolutionary design: CA, L-systems, emergence, ...

Circuit Evolution and EHW

Two views towards EHW: the design optimisation view and adaptive hardware view.

1. Design view: some circuits are too difficult or costly to design using the conventional approach;
2. Adaptive hardware view: circuits should be able to adapt online autonomously after they are produced.

Some Key Issues in EHW

Scalability: Can we solve large problems efficiently?

1. In terms of chromosome representation — direct representation
2. In terms of evolution — the computational complexity issue

Will the electronic speed help us?

Reliability: Can we ensure the reliability of the EHW?

Molecular Design

Forward problem: From molecules to their properties using nonlinear mapping, e.g., ANNs.

Backward problem: From properties to search for molecules using a discovery tool, e.g., EAs.

Key Issues

1. Chromosome representation
2. Search operators
3. Fitness evaluation
4. Hybridisation (knowledge lean + knowledge rich methods)

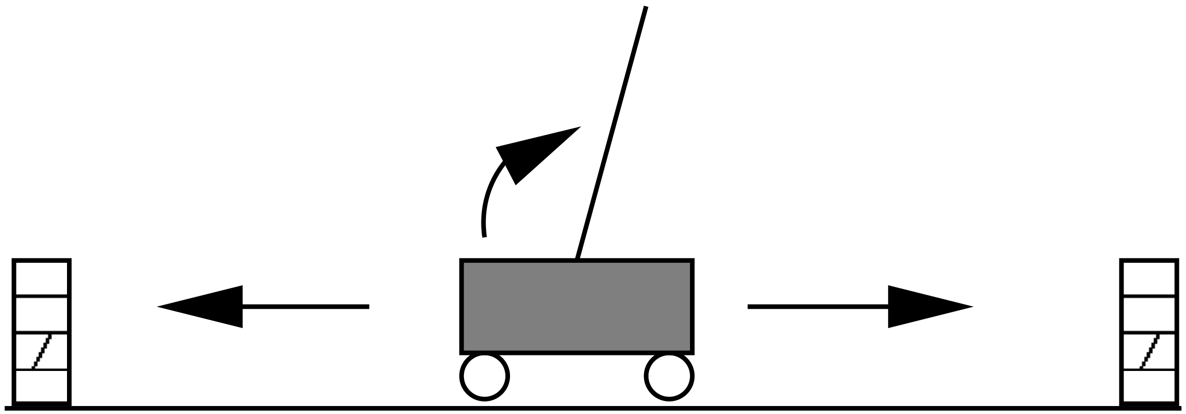
Exam Structure

Undergraduates and conversion MSc: Five questions (15, 15, 20, 25, 25).

MSc in NC and ACS: Four questions (10, 15, 25, 50).

An Example on a VERY Different Topic

The cart-pole problem, as shown in Figure 13 is illustrative of many of the features of more complex control problems. The task is to maintain the pole which is attached by a hinge to the top of the cart balanced for a set amount of time without ever pushing the cart over the end of the finite tracks ($\pm$ 1 meter). The pole may only deviate from the vertical by a prescribed number of degrees ($\pm$ 12 degrees). The task is complicated by only allowing a discrete force to be applied to the cart in either direction ($\pm$ 10 Newtons). It is further complicated by starting the system from a randomly chosen position ($\pm$ 0.5 m, $\pm$ 0.1 m/s, $\pm$ 2 deg, $\pm$ 1 deg/s). (Other factors, such as friction etc., are ignored.)



In essence, the task here is to determine whether the cart should move to left or right according to four input variables, i.e., the position (x) and speed (x') of the cart and the angle (θ) and speed (θ') of the pole.

Assuming that the controller we are going to design is rule-based. Use co-evolution to evolve the rule-based controller. Please describe the detailed design of such a co-evolutionary system. In particular, the following are required (Please explain your design choices):

1. The genotypic representation of an individual.

2. The phenotypic representation of an individual.
3. The fitness evaluation function for an individual.
4. Evolutionary operators, such as crossover and mutation, used in the co-evolutionary system.
5. The selection scheme.
6. As the last step of your design, please analyse the strength and potential problems of your design.

You may use any techniques you have learned, e.g., any niching and/or speciation techniques.

A Possible Answer

genotypic representation There are a number of design decisions to be made based on the problem description.

- To avoid the difficult problem of credit assignment (at least partially), the Pitt approach is used here. Each individual represents a rule-based controller for balancing the pole. The number of rules in an individual may be fixed or variable.
- There are four continuous inputs and one binary output for this problem. The continuous inputs can be discretised because the action for very close values are likely to be the same, i.e., the action for $x = 3.14159$ is very likely to be the same for $x = 3.1416$ provided other values are also similar. Hence we can divide the 4-d continuous space into boxes. For each variable (i.e., along each dimension), there are

several different regions, such as $x \leq x_1, x_1 < x \leq x_2, x_2 < x \leq x_3, x > x_3$. This defines four regions along the x axis.

- After the above discretisation, each rule can be represented by five genes: the first four describe which regions the current input vector is in and the last one indicates the action to be taken. In order to find appropriate values of $x_1, x_2, \ldots$, they can be evolved as well. Hence, each individual in our evolutionary system consists of a set of rules and a real-valued vector.

phenotypic representation This is the same as the genotypic representation. In other words, there is an one-to-one mapping between them.

fitness evaluation This is where we use niching techniques and co-evolution in the system. In order to encourage the formation of different niches so that they can be combined to form a better integrated rule-based controller, fitness sharing can be used. The fitness of an individual consists of two components. The first component (f_1) depends on the average time the individual balances the pole from a random starting position. The longer the time, the higher f_1. The second component (f_2) depends on whether the individual can balance the pole from a state (i.e., the four variable vector) that no other individuals in the current population can balance the pole successfully. Two states are considered the same if they are in the same box. The more such states, the higher f_2. The fitness of an individual, i, is thus defined as

$$f(i) = f_1(i) + \alpha f_2(i)$$

where α is a coefficient. To encourage fitness sharing and co-evolution, $\alpha > 1$ in most cases.

evolutionary operators Both crossover and mutation are used. Both rules and the real-valued vector (for determining regions) can be evolved/modified.

- The crossover between two individuals' rule sets is as follows: select two individuals based on their fitness (the selection scheme will be described next) and swap a subset (chosen uniformly at random) of rules between them.
- Mutation of a rule set changes outputs of some rules (chosen uniformly at random).
- Mutation of a real-valued vector is achieved by adding a Gaussian noise to each component of the vector independently. Self-adaptation can be used to adjust the variance of Gaussian mutations. Crossover could also be used for real-valued vectors although its not necessary here.
- Other operators could be used as well.

selection scheme Rank based selection is used here because it is able to maintain a constant selection pressure throughout an evolutionary process. Tournament selection is also acceptable.

strength and potential problems The co-evolutionary system described above have the following major features.

- Co-evolution and fitness sharing encourages formation of different species. They also help to maintain diversity in the population. Such diversity will make the evolution of a good controller easier because we can combine good subsets of rules together.
- The potential problem with the system is to get the coefficient α right. α plays a critical role in balancing individual's own performance and its cooperation (through fitness sharing) with others. A small α will not be able to achieve the goal of fitness sharing since individuals will be very similar to each other. A large α will make individuals

disregard their performance and concentrate on pole-balancing from a single position, because there is little incentive to do more.