EQML- An Evolutionary Qualitative Model Learning Framework

Wei Pang and George Macleod Coghill
School of Engineering and Physical Sciences, King's College,
University of Aberdeen, Aberdeen, AB24 3UE, UK
Phone: +44-1224-273829, Fax: +44-1224-272455
Email :{wpang, gcoghill}@csd.abdn.ac.uk

ABSTRACT: In this paper, an Evolutionary Qualitative Model Learning Framework (EQML) is proposed and tested by learning the qualitative metabolic models under the condition of incomplete knowledge. JMorven, a fuzzy qualitative reasoning engine, is slightly modified and integrated into the framework as a sub module to represent and verify the learnt models. Three metabolic compartment models are tested by two evolutionary algorithms (Genetic Algorithm and Clonal Selection Algorithm) in EQML. Finally the efficiency of these two algorithms is evaluated.

KEYWORDS: Model Learning, Evolutionary Computing, Genetic Algorithm, Clonal Selection Algorithm, System Identification, Artificial Immune Systems

1. INTRODUCTION

Biological data are often sparse and imprecise because of technical limitations and characteristics of biological systems. Qualitative Analysis approaches including Qualitative Simulation and Qualitative Inference become essential especially when dealing with complex biological systems.

Better understanding metabolic systems is crucial in biology, in [5] a coupled two-compartment metabolic model was qualitatively learned by QSI-ILP algorithm, and the minimum required training data (Kernel Sets) for learning right models were studied.

However, in biological experiments, the available experimental data do not always include the Kernel Sets. Under this condition, the learning task becomes very difficult: There will be many qualified models which are consistent with the experimental data, and we have to find as many these models as possible in a big search space, because the right models are among these qualified models. In this paper, we will use evolutionary approaches to deal with this problem. Three two-compartment models were analyzed as test examples in our work. Furthermore, we focus on investigating the learning ability of evolutionary algorithms under the above situation.

The rest of this paper is organized as follows: Section 2 introduces the background knowledge. Related work is briefly reviewed in Section 3. EQML framework is detailed in Section 4. Section 5 shows the experimental results. Finally the conclusions are drawn and future work is discussed in Section 6.

2. BACKGROUND KNOWLEDGE

2.1 QUALITATIVE SIMULATION

Qualitative Simulation [11] is a system simulation technique which can predict the qualitative behaviours of real world physical and biological systems. These systems are modelled by Qualitative Differential Equations (QDEs). JMorven [2], one of the Qualitative Simulation engines, is a Java implementation of Morven [4]framework. It extends the functions of Qualitative Simulation in the following aspects:

It incorporates the fuzzy theory into Qualitative Simulation, which enables it to perform fuzzy qualitative reasoning; it introduces the fuzzy vector envisionment [4]that can reason about more than two derivatives of a variable, which makes JMorven more flexible; In addition, the utilization of parallel techniques makes JMorven more efficient. All the above advantages make JMorven a better choice as a verification component in our framework.

2.2 QUALITATIVE MODEL LEARNING

Qualitative model learning (QML), also called qualitative system identification, can be viewed as the inverse of qualitative simulation. It involves inferring the qualitative structures (namely QDEs) of the complex systems from given qualitative data (often incomplete and imprecise). The meaning of the word "Qualitative" is twofold: First, the experimental data used are qualitative, which may be transformed from quantitative data or be qualitative nature. Second, the model structures that we infer are also qualitative.

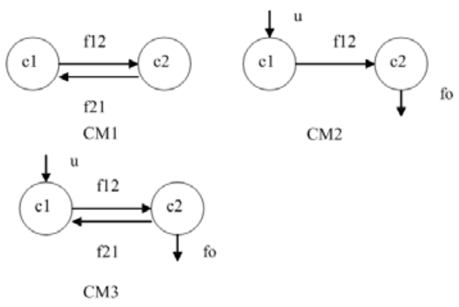
2.3 EVOLUTIONARY COMPUTATION

Evolutionary Computation has been successfully applied to solve many complex problems. It can be applied to achieve QML, because it has domain independent representation, easy-to-implement quality and high searching efficiency.

In this paper, Genetic Algorithm(GA) and Clonal Selection Algorithm (CSA) [6] are tested. The main difference between GA and CSA is, instead of two main genetic operators: crossover and mutation in GA, CSA is inspired by the clonal selection principle [7] of immune system, and uses hyper-mutation and re-selection to implement the search process. In performing different tasks, CSA and GA may exhibit different performance.

2.4 COMPARTMENTAL MODELS OF METABOLIC SYSTEMS

Metabolic systems are often modelled by two-compartment models(See Figure 1). In the compartmental model, if input u and output *fo* do not exist, the model becomes a coupled closed system, denoted as model CM1 in this paper. CM2, CM3 are defined in a similar way. Table 1 shows the QDE and JMorven Description for CM3.



QDE	JMorven Differential Plane 0
f12=M+(c1)	func (dt 0 f12) (dt 0 c1)
f21=M+(c2)	func (dt 0 f21) (dt 0 c2)
fo=M+(c2)	func (dt 0 fo) (dt 0 c2)
qx=f12 - f21	sub(dt 0 qx)(dt 0 f12)(dt 0 f21)
c1'=u - qx	sub(dt 1 c1)(dt 0 u)(dt 0 qx)
c2'=qx - fo	sub(dt 1 c2)(dt 0 qx)(dt 0 fo)

Figure 1: Two-compartment Metabolic Models.

Table 1: QDE and JMorven Description for CM3

3 RELATED WORK

Many QML systems have been proposed in the last two decades. Among them the following are noteworthy: GENMODEL [8], QSI [13], MISQ [12], and QSI-ILP [3]. However, none of the above used the evolutionary approach. All of them were based on straightforward logic reasoning approaches, which mainly depended on domain knowledge to find models. Using QSIM description to represent models, none of them can easily incorporate fuzzy theory. None of them except QSI-ILP can produce causal ordered models [9], which are physical or biological meaningful.

All of the above become the motivation of developing an evolutionary qualitative model learning system to explore the potential ability of evolutionary approaches in the field of QML.

4 EQML FRAMEWORK

The proposed EQML framework is composed of three phases: constraint generation, constraint filter, and evolutionary search. See Figure 2 for details.

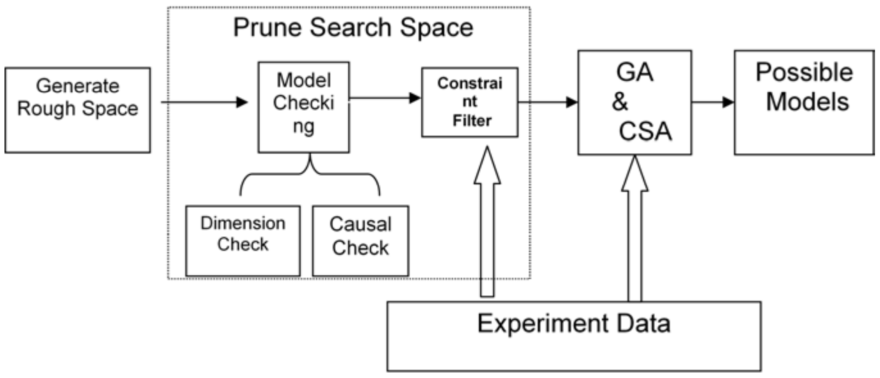


Figure 2: EQML Framework.

4.1 CONSTRAINT GENERATION

The constraint generation is similar to GENMODEL [8] except performing an additional dimensional check [1] and a Causal Check [9]. In this phase, given all the observed variables, number of derivatives for each variable, range and dimension (if available) for each derivative, and possible constraint types [10] (monotonically increasing, monotonically decreasing, and algebraic constraints), the constraint generator will generate all the possible constraints. The power set of the generated constraint set ***CS*** constructs the rough problem space ***RS***.

4.2 CONSTRAINT FILTERING

Second, all the constraints in CS will be checked for consistency by the constraint filter. Only the constraint which is consistent with the given qualitative behaviors will be preserved. The power set of all the remaining constraints constructs a refined search space RSS. Given complete behaviors of the systems, RSS will have the minimum size; otherwise, RSS may be very large.

4.3 EVOLUTIONARY SEARCH

As we have mentioned before, under the condition of incomplete training data set which does not include the Kernel Set, there will be many qualified models. The landscape of ***RSS*** will have many "equally high" peaks, and the size of ***RSS*** may become very large. In our work, Genetic Algorithm (GA) and Clonal Selection Algorithm (CSA) [6]are employed to find as many these peaks as possible in ***RSS***. The details of these two algorithms will be described in the following subsections.

4.3.1 Individual Encoding

Each individual (chromosome in GA and antibody in CSA) in the population (or antibody repertoire in CSA) indicates a candidate model and is encoded as a 0-1 binary vector. "1" denotes the corresponding constraint is a component of the model, and "0" otherwise.

4.3.2 Model Evaluation

The fitness function (or affinity function in CSA) of each individual will be determined by evaluating the model that this individual represents. In [3], some criteria have been set up to evaluate a candidate model. In our work, the following properties of a candidate model will be considered in sequence: redundancy, completeness, Contradiction, Connection, Causality, Coverage, and Model Size. Suppose M1 and M2 are two models, the Boolean function of comparing these two models is defined as follows:

```
CompareModel (M1, M2)
{
        1.Redundancy Comparison:  Calculate the RedundentRate r1,r2 of M1  and M2 respectively;
           If (r1 == r2 == 0) Goto Step 2;  //Both M1 and M2 are not redundant, go to next comparison.
           If (r1 > r2) return false; //M2 is better than M1
           else if (r1 < r2) return true;  //M1 is better than M2
           else if (r1 == r2) //M1 and M2 are equally "bad"
              return a random Boolean variable;
        (Step 2-7 are similar to Step 1)
        2. Completeness Comparison: Calculate the CompleteRate
        3. Contradiction Comparison: Calculate ContradictionRate
        4. Connection Comparison: Calculate Is_Connected
        5. Causality Comparison: Calculate Is_CausalOrdered
        6. Coverage Comparison: Calculate coverageRate
        7. Model Size Comparison
}
```

For each property a measured variable is used to evaluate the quality of the model, such as the above *RedundentRate* and *CompleteRate*. The corresponding functions have been developed to calculate the value of these measured variables. Step 7 is optional, which indicates the Minimum Description Length (MDL) principle. In Step 1, the *RedundentRate* is defined as:

$$\text{RedundentRate} = \text{Num\_Redundent\_Cons}/\text{Num\_All\_Cons} \qquad (1)$$

Here *Num_Redundent_Cons* is the number of the redundant constraints; *Num_All_Cons* is the number of all the constraints in the model. In Step 2, the *CompleteRate* is defined as:

$$\text{CompleteRate} = \text{Num\_Model\_Var}/\text{Num\_All\_vars} \qquad (2)$$

Here *Num_Model_Var* is the number of the variables and their derivatives included in current model, *Num_All_vars* is the number of all the given variables and their derivatives. If a model is complete, which means it includes all the observed variables and their derivatives, the *CompleteRate*=1.0, otherwise it will be a number between [0, 1). In Step 6, the *CoverageRate* is defined as:

$$\text{CoverageRate} = 1 - \text{Data\_Not\_Included}/\ \text{training\_Data} \qquad (3)$$

Data_Not_Included is the number of the behaviors which are in the given training data set but are not included in current model. *Training_Data* is the number of all the given training data. The embedded JMorven sub module is used to get all the behaviors of this model by performing a total envisonment [4], then the *CoverageRate* can be calculated by comparing the difference between the behaviors of current model and the training data.

Other measured variables can be defined in a similar way. It should be pointed out that the calculation order of the above seven steps is based on the syntactic, semantic considerations of the model and computation efficiency. The Coverage test should be placed in the last stage because JMorven simulation is expensive in computation. Only the model that passes the first five checking processes is worth simulating.

4.3.3 Genetic Algorithm

It is difficult to assign a real value to the fitness of the individuals. But the rank of the individuals can be set up by the above function *CompareModel*, so the tournament selection is adopted in GA.

Three crossover operators: single-point, two-point and uniform crossover were tested and the two-point crossover was proven to be the most appropriate one in our problem. The mutation operator is single-point mutation.

To keep the best models throughout generations, an elitism strategy is used, that is, a memory pool is created to record all the individuals that pass the first six checking steps of *CompareModel*. In each generation, some randomly selected individuals will be replaced by the individuals in the memory pool.

4.3.4 Clonal Selection Algorithm

Given incomplete knowledge, there may be many qualified models, so the search space ***RSS*** has some similarities to that of multimodal functions. The optimization version of the CSA [6] has been proven to be an efficient algorithm in multimodal function optimization problems, so it is applied to search ***RSS***.

In the optimization version of CSA, all the antibodies in the antibody population are expected to find different optima. In each generation, all the antibodies will be selected and cloned with the same clonal size independently, forming a temporary population. Then the temporary population will suffer the hyper-mutation process, but there must be at least one antibody unchanged in each cloned sub-population to keep the best solution. Finally the next generation population will be generated by performing a re-selection operator on the temporary population.

In our work, to enhance the ability of finding more qualified models, we add an additional operation in CSA: In each generation, when an antibody finds a qualified model, it will be recorded and replaced by a randomly generated antibody.

5 EXPERIMENTAL METHODOLOGY AND RESULTS

The above mentioned three metabolic models are tested by EQML. First, the total envisionment for each model is performed by JMorven, then the complete qualitative behaviors of these models are obtained. The qualitative values that all the variables can take are *negative*, *zero* and *positive*. (The quantity space of the variables is represented by fuzzy tuples, see [2] for details). For simplicity, the input *u* is assumed to be steady positive in CM1 and CM2. We make the assumption that there are no noise, no unmeasured variables and no missing dimensions in the training data. There are 6 qualitative states in CM1 (See Table 2), 14 states in CM2 and 16 states in CM3. For each model, the power set ***R*** of the states is generated. For example, there will be $2^6 - 1$ elements in ***R*** for CM1 model.

	1	2	3	4	5	6
f12	zer	zer	pos	pos	pos	pos
c2	{zer , zer}	{pos,neg}	{zer ,pos}	{pos,zer}	{pos,neg}	{pos, pos}
c1	{zer , zer}	{zer , pos}	{pos ,neg}	{pos,zer}	{pos,pos}	{pos, neg}
f21	zer	pos	zer	pos	pos	pos
qx	zer	neg	pos	zer	neg	pos

Table 2: Qualitative States of CM1 (zer= ”0”, pos= ”+”, neg= ”-”)

Each element *t* in R is considered as a training data set to construct an experiment. In each experiment, the first two phases of EQML are performed to get ***RSS***. Whether to perform the third phase depends on the complexity of ***RSS***, because the evolutionary approach will not perform well in small search space.

In CM1 and CM2 experiment, all the elements in R which have two qualitative states have been tested. Some difficult experiments (with large ***RSS***) are selected out to carry on the third phase of EQML. The GA and CSA in this phase will run ten times and the average performance has been recorded.

In our work, the landscape of the search space is very complicated, resulting from the selection of difficult experiments with large ***RSS***. Since our task is to find as many optima as possible, the parameters of GA and CSA should be adjusted to search in the multi-peak landscape effectively.

In GA, we maintain a population with 1000 individuals, and use the tournament selection of size 2 to keep the selection pressure low, avoiding the premature convergence. In the experiments, if the mutation probability is too small, GA will find only a few optima. In order to find more optima, a high mutation probability value 0.1 is used to enhance the explorative capability of individuals and proven by the experimental result to be appropriate. The crossover probability is 0.8. In CSA, classical parameter values are taken: the number of the antibodies is 1000, the clonal size is 10, and the hyper-mutation probability is 0.1. The running time of both algorithms is 60 seconds. Table 3 shows the experimental results, more detailed information can be found in www.csd.abdn.ac.uk/~wpang/Compartment .

Experiment ID	Number of Qualitative States	Search Space	Models Found (GA/CSA)	Generations (GA/CSA)
CM1-17	2(1,5 in Table 2)	16,777,215	7.2 / 9.2	152.11/50.7
CM1-33	2(1,6 in Table 2)	67,108,863	4.0 /4.6	282.00/44.8
CM2-9	2	16,777,215	2/5.6	383.3/65.5
CM2-10	2	268,435,455	3.9/1	214.7/48.3
CM2-40	2	67,108,863	11.4/6.9	81.8/58.0
CM3-39	4	33,554,432	22.8/29.7	43.2/44.2
CM3-51	4	33,554,432	25.2 /27.4	45.6/41.6
CM3-15	4	2^{36}-1	15.6/1.1	118.5/20.9

Table 3: Experimental Results

From the experiment we can see that CSA performs slightly better than GA in middle-sized search space such as CM2-9, CM3-39, but GA will surpass CSA in large-sized search space such as CM2-10 and CM3-15. It should be pointed out that the algorithms cannot always find the right models under incomplete training data, so we only focus on the search ability of the algorithms.

6 CONCLUSIONS AND FUTURE WORK

In this paper, an evolutionary learning framework to study qualitative models under the circumstances of incomplete knowledge is proposed. A method for evaluating the fitness of the individuals has been designed and applied in the evolutionary algorithms. Two evolutionary algorithms: Genetic Algorithm and Clonal Selection Algorithm are employed to find the possible models in some difficult situations. In addition, JMorven is used to test the validity of the models.

The characteristics of our framework are: first, evolutionary approaches are adopted, resulting in the easy implementation of the framework; second, the integration of JMorven enables the framework to deal with fuzzy qualitative data easily; third, the framework can find causal ordered models which are physical or biological meaningful; finally, three metabolic compartment models are analyzed and tested as benchmark problems by GA and CSA. Some meaningful results have been obtained from the experiments.

Future work will involve handling the noisy data, finding unmeasured variables, and investigating the potential ability of evolutionary approaches in more complex hybrid biological systems.

REFERENCES

[1] R. Bhaskhar and A. Nigam, 1990, “Qualitative physics using dimensional analysis”, Artificial Intelligence Vol.45: pp.73 - 111.

[2] A. M. Bruce and G. M. Coghill, 2005, “Parallel fuzzy qualitative reasoning”, Proceedings of the 19th International Workshop on Qualitative Reasoning, Graz, Austria, pp. 110 - 116.

[3] G M. Coghill, S M. Garrett, A Srinivasan,and R D. King, 2006, “Qualitative system identification from imperfect data”, Technique Report AUCS/TR0501, Department of Computing Science, University of Aberdeen.

[4] George M. Coghill, September 1996, Mycroft: “A Framework for Constraint based Fuzzy Qualitative Reasoning”, PhD thesis, Heriot-Watt University.

[5] George M. Coghill, S.M. Garrett, and R D. King, 2004, “Learning qualitative metabolic models”, European Conference on Artificial Intelligence (ECAI’04), Valencia, Spain, pp. 445-449.

[6] de Castro and Von Zuben, July 2000, “The clonal selection algorithm with engineering applications”, Proceedings of GECCO, Workshop on Artificial Immune Systems and Their Applications, Las Vegas, USA, pp. 36-39.

[7] F.Burnet, 1959, “The Clonal Selection Theory of Acquired Immunity”, Cambridge University Press, Cambridge.

[8] D. T. Hau and E. W. Coiera, 1993, “Learning qualitative models of dynamic systems”. Machine Learning Vol. 26, pp. 177-211.

[9] Y. Iwasaki and H. A. Simon, 1986, “Causality in device behavior”. Artificial Intelligence Vol 29, pp.3-32.

[10]Benjamin Kuipers, 1986, “Qualitative simulation”, Artificial Intelligence, 29: pp.289-338.

[11]Benjamin Kuipers, 1994, “Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge”, MIT Press, Cambridge, MA.

[12]Bradley L. Richards, I. Kraan, and Benjamin Kuipers, 1992, “Automatic abduction of qualitative models”, National Conference on Artificial Intelligence, pp. 723-728.

[13]A.C. Cem Say and Selahattin Kuru, 1996, “Qualitative system identification: deriving structure from behavior”, Artificial Intelligence, Vol. 83, pp.75-141.