Automatic reactor model synthesis with genetic programming
David J. Dürrenmatt and Willi Gujer

ABSTRACT
Successful modeling of wastewater treatment plant (WWTP) processes requires an accurate
description of the plant hydraulics. Common methods such as tracer experiments are difficult and
costly and thus have limited applicability in practice; engineers are often forced to rely on their
experience only. An implementation of grammar-based genetic programming with an encoding to
represent hydraulic reactor models as program trees should fill this gap: The encoding enables the
algorithm to construct arbitrary reactor models compatible with common software used for WWTP
modeling by linking building blocks, such as continuous stirred-tank reactors. Discharge
measurements and influent and effluent concentrations are the only required inputs. As shown in a
synthetic example, the technique can be used to identify a set of reactor models that perform equally
well. Instead of being guided by experience, the most suitable model can now be chosen by the
engineer from the set. In a second example, temperature measurements at the influent and effluent
of a primary clarifier are used to generate a reactor model. A virtual tracer experiment performed on
the reactor model has good agreement with a tracer experiment performed on-site.

Key words | grammar-based genetic programming, hydraulic reactor systems, modeling, operating
data

INTRODUCTION
A key element for successful wastewater treatment plant (WWTP) modeling is an accurate description of the hydrau-
lic processes. In water technology, transport and mixing phenomena can often be approximated sufficiently by cas-
cading ideal reactors such as the continuous-stirred tank reactor (CSTR) and plug-flow reactor (PFR) models (Alex
et al. 2002; Gujer 2008).

The construction of an appropriate model given influent and effluent observations is a widely studied system identifi-
cation problem and many techniques exist (see, e.g. Ljung 1987; Keesman 2011). However, system identification remains
a difficult task particularly when the structure of the system is unknown (Flores & Graff 2005). In systems analysis for
water technology, two methods for model identification prevail (Gujer 2008). The first method consists of the analysis of an experimentally determined impulse response (e.g. by a tracer experiment) that gives insights into mixing phenomena and that can, for some few ideal reactor systems, also give insights into model structure (Keesman & Stigter 2002; Gujer 2008). From a practical point of view, however, tracer experiments are labor-intensive and are hindered by high flow variability, required mixing lengths and density effects; therefore they are often omitted.

The second method consists of the process of manually adjusting model structure and parameters until a pre-defined
objective function is minimized. Although this procedure is widely accepted, it bears some problems. Still, extensive
measuring campaigns may be required and manual search by trial and error is inefficient as it only covers a small
part of the model and parameter spaces, thus the most appropriate model is not guaranteed to be found.

As a consequence, engineers often rely on experience and intuition only, which is dissatisfying because inappropri-
ate hydraulic modeling can have a significant impact on predicted effluent concentrations (Metcalf & Eddy Inc.
2003).

In this paper, a highly practical method is presented that provides the engineer with a small set of hydraulic reactor
models that perform equally well; the most suitable model can then be selected by taking expert knowledge into
account. The set of models is generated using genetic programming (GP), an advancement of genetic algorithms.

GP is an evolutionary computational technique used for optimization and particularly suited for complex problems with high-dimensional search spaces (Koza 1992) and has previously been considered for system identification, mainly for symbolic regression of sets of ordinary differential equations (for an overview, see Flores & Graff 2005). For the implementation presented in this paper, in contrast, a special GP tree encoding was defined that encodes hydraulic reactor models. The encoded models consist of the building blocks CSTR and PFR and are therefore compatible with common software used for WWTP modeling. As input, the method requires easily measured signals only (e.g. temperature data), even if they are influenced by physical and chemical reactions, in addition to discharge data.

Two alternative approaches for the direct generation of reactor models worth mentioning have used genetic algorithms (Laquerbe et al. 2001) or found a model by simplifying initially complex super-structures (Hocine et al. 2008). Both approaches, however, require the measurement of the residence time distribution.

MATERIAL AND METHODS

Reactor modeling

The reactor types commonly used to model hydraulic characteristics are the CSTR and PFR. By linking these basic reactors serially or in parallel, even complex situations can be modeled. Each reactor has one or more in- and out-flows. (A reactor with more than one outflow can be imagined as a reactor with a subsequent flow divider.)

In this study, only one compound is considered and its concentration is calculated for each reactor. The relevant elements for the construction of arbitrary reactor models are the inflow and outflow nodes, CSTRs and PFRs and reactions and fluxes. An example is given in Figure 1.

Inflow and outflow node

Each hydraulic reactor system has an inflow node that acts as a junction node to combine the system influent (with flow $Q_{in}$ and concentration $C_{in}$) with zero or more return flows. Similarly, the outflow node collects fluxes and releases them from the system. Between the inflow and outflow nodes, there is any number of arbitrarily linked reactors.

CSTRs and PFRs

The CSTR, probably the most important building block, is implemented as an ordinary differential equation (ODE). It is assumed that its volume $V$ is constant over time; the mass balance for compound $C$ in the reactor can therefore be written as

$$\frac{dC}{dt} = \frac{1}{V} \left( \sum_j Q_j C_j - C \sum_j Q_j \right) + r$$

(1)

with one or more inflow $Q_i$ and a reaction term $r$. PFRs are implemented as a delay without reaction (as delays are typically very small), and the effluent concentration at time $t$ is

$$C_t = C \left( t - \frac{V}{\sum Q_j} \right).$$

(2)

Reactions

Reactions taking place in the CSTRs are modeled with zero-order ($r = KC$ with reaction coefficient $K$), first-order ($r = kC$ with reaction coefficient $k$) or Monod kinetics ($r = q_{max}C/(K_S + C)$) with maximal activity $q_{max}$ and half-saturation coefficient $K_S$. Reactions taking place in the PFRs are not modeled. However, approximating the PFR by a cascade of $n$ CSTRs is an option because plug-flow behavior is approximated for $n \to \infty$.

Fluxes

Because the reactor volumes are constant, the flow in each link can be calculated by solving a linear equation system given the total inflow to the system and, if there are any reactors with more than one effluent, given their flow ratios expressed by the quotient of the weight factors assigned to each of the links. The system depicted in Figure 1, for instance, has unknown flows $Q_{l1}$, $Q_{l2}$, ..., $Q_{l5}$ while the
total inflow $Q_{\text{in}}(t)$ as well as the two weight factors $w_{L4}$ and $w_{\text{out}}$ are given. This allows us to write the following linear equation system consisting of four mass balance equations

\begin{align*}
Q_{L1} &= Q_{\text{in}} \tag{3} \\
Q_{L1} + Q_{L5} &= Q_{L2} \tag{4} \\
Q_{L2} &= Q_{L3} + Q_{L4} \tag{5} \\
Q_{L4} &= Q_{L5} \tag{6}
\end{align*}

and one equation that expresses the ratio of the effluent flows of reactor ‘CSTR 2’

\[ \frac{Q_{L3}}{Q_{L4}} = \frac{w_{\text{out}}}{w_{L4}}. \tag{7} \]

**Genetic programming**

GP is a search algorithm inspired by nature (Koza 1992). It aims to evolve mathematical expressions or computer programs by mimicking biological evolution. Starting with a population of individuals (random programs), new generations are bred. During each generation, the fitness of every individual is evaluated by a fitness function. The fittest ones among the population are then more likely to survive to the next generation. They can be copied unaltered (reproduction), they can feature random changes (mutation), or they can be used to generate new offspring by combining two parents (crossover). This process is repeated until either a given fitness criterion is met or a maximum number of generations is reached.

The evolved, tree-like computer programs can vary in length, which is a valuable characteristic of GP. Other advantages include the absence of a tendency for the entire population to converge and the fact that the form of the solution does not need to be known in advance (Tsakonas 2006). A function set and a terminal set from which GP can choose to build the programs, in addition to a measure of fitness, need to be specified. In this paper, a grammar-based paradigm is selected with a context-free grammar that consists of a set of terminal nodes, function nodes, a set of reproduction rules that define for each function the possible child function(s) and a starting symbol (the root of the tree). The definition of a grammar avoids the generation of meaningless programs and thereby significantly reduces the search space.

**Tree encoding**

A tree encoding was defined to represent hydraulic reactor models as computer programs. The functions and terminals available for the program are listed in Table 1, and the grammar rules are given in Table 2. Every program starts with a ROOT function, which has two child nodes, that splits the program into two branches. The left branch (starting from the ADF_L function) encodes the layout of the model (the reactors and their connections), whereas the right branch encodes the reaction rate, which can be referenced by some of the reactors of the model. The program tree is
A recursive decoding starting from the outermost terminals. Once the decoding reaches the ROOT function, the inflow and outflow nodes are added, resulting in an object-oriented representation of the hydraulic model. The decoding is illustrated in Figure 2.

### Fitness function

The fitness $F$ of a program tree is a scalar value and is evaluated by a fitness function that takes into account the error between the actual and predicted effluent and the complexity of the model. The consideration of model complexity is important because tree sizes tend to grow during evolution (Koza 1992; Soule & Foster 1998), consequently leading to more complex models that have limited generalization ability, i.e. that cause over-fitting of the data (McKay et al. 1997). However, over-fitting can be prevented by introducing a term for ‘parsimony pressure’ that penalizes large tree structures (Soule & Foster 1998). Thus, the fitness value for an individual is calculated as

$$F = CV(RMSD) + aL$$

where $CV(RMSD)$, the coefficient of variation of the root mean square deviation, expresses the error when comparing the prediction of the numerically solved model with the measured series, $a$ is the coefficient for parsimony pressure and $L$ the number of links in the model. Because lower residuals represent better models, the goal is to minimize the value calculated by the fitness function. If a reactor model cannot be solved, an infinite fitness value is returned. This discourages its survival into the next generation.

### Selection, crossover and mutation

During evolution, the fittest individuals of each generation are selected for the next generation in tournaments. In each tournament, a number of individuals are picked at random. The best is chosen with selection probability $p$, the second best with probability $p(1-p)$, etc. Tournament selection allows an easy adjustment of the selective pressure.
by the tournament size parameter. Some of the selected individuals experience random mutations, where nodes of the program tree are randomly exchanged. Others are affected by random crossovers, where two individuals exchange parts of their programs. The selection probability and the rates of crossover, mutation and reproduction remain constant during breeding to avoid the convergence of the population and thus getting stuck in a local optimum. The maximum tree depth is constrained to avoid the generation of overly complex models and is therefore another means to prevent over-fitting, next to parsimony pressure introduced in Equation (8).

**Procedure**

The search algorithm is run several times in parallel to obtain a palette of equally well-performing models. Theoretically, the global optimum will always be found. However, breeding is stopped when a pre-defined fitness criterion is satisfied or when the maximum number of generations has been reached. In addition, due to the noisy nature of measurements, the use of different input data for the individual runs is encouraged. Eventually, the model palette can then be presented to the expert, who selects the most appropriate model.

**RESULTS AND DISCUSSION**

**Synthetic system: CSTRs in series with and without reaction**

The power of the proposed method was first assessed by investigating its ability to identify a predefined synthetic reactor model in two experiments. The reactor model consisted of two CSTRs in series (volumes of 400 m$^3$ and 1,600 m$^3$). In the first experiment, no reaction took place in any of the two reactors, whereas in the second experiment, a degradation reaction occurred in the second reactor. The reaction had first-order kinetics with $k = 1.2$ h$^{-1}$. Three days of artificial influent flow and concentration data were generated from sine waves superposed with an Ornstein–Uhlenbeck process (the OU-process can be considered as the continuous version of the discrete first-order auto-regressive process, AR(1)) ([Uhlenbeck & Ornstein 1930](#)). No measurement error was assumed. The time series had an average of 360 m$^3$/h and 16.7 g/m$^3$, respectively.

For each experiment, three populations were bred, and only the total volume of all reactors was made available to the algorithm. The parameter values used for the evolution are given in Table 3. The reactor models of the fittest individual of every population are shown in Figure 3. In all breeds of Experiment 1, evolution was stopped because the fitness criterion was met. Notably, the best individuals share the fitness value although their schemes differ. In Experiment 2, in all but one case, breeding was stop because the fitness criterion was satisfied. It is now the engineer’s task to select the preferred system, taking into account the performance of the system while also considering its simplicity in addition to background knowledge not provided to the GP algorithm.

Lack of identifiability is probably the main challenge when inferring reactor models; both the arrangement of the reactors and the parameter values might be unidentifiable. Under certain circumstances (e.g. a lack of reactions), different arrangements can lead to exactly the same effluent concentrations. This dynamic is visible in the results of the first experiment, in which the arrangement of the reactors cannot be identified. Possible countermeasures include constraining the search space by providing additional knowledge about the system, or using the concentrations within the reactor to construct a more elaborate fitness evaluation. Due to the reactions in the second CSTR of Experiment 2, the arrangement of the reactors was correctly identified.

The fact that in Breed #3 of Experiment 2, breeding was stopped by reaching the maximum number of generations indicates that the optimal solution has not yet been found. It is therefore advisable to investigate the population statistics. Figure 4 shows that, although the minimal fitness decreases over time, a significant number of unfit individuals are still being introduced and a high diversity is maintained. A high diversity in the population increases the chance that mutations and crossovers lead to new, stable individuals.

### Table 3: The genetic programming parameters used for the experiments

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population size</td>
<td>500</td>
<td>Tournament size</td>
<td>7</td>
</tr>
<tr>
<td>Max. generations</td>
<td>100</td>
<td>Selection probability</td>
<td>0.8</td>
</tr>
<tr>
<td>Fitness criterion</td>
<td>$F &lt; 5 \times 10^{-4}$</td>
<td>Crossover rate</td>
<td>0.50</td>
</tr>
<tr>
<td>Parsimony pressure</td>
<td>$a = 10^{-4}$</td>
<td>Mutation rate</td>
<td>0.49</td>
</tr>
<tr>
<td>Min. tree depth</td>
<td>4</td>
<td>Reproduction rate</td>
<td>0.01</td>
</tr>
<tr>
<td>Max. tree depth</td>
<td>18</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
superior individuals. Consequently, it would be a matter of time until the global optimum was found if breeding was continued.

Simulation of a tracer experiment

The proposed technique was used to generate a reactor model for a primary clarifier (V = 750 m³) in a mid-sized Swiss WWTP and to investigate to what extent virtual tracer experiments performed on the generated models agree with a tracer experiment performed on-site. Influent and effluent temperature was measured for four days, the total WWTP influent flow data was extracted from the process information system of the plant.

Reactor models were obtained in three breeds without constraining the total reactor volume and using the temperature time series as well as the parameters given in Table 3. The best-performing models were used to simulate a tracer experiment, the results of which were compared to a tracer experiment performed on site (Figure 5).

The evolved systems model the measured temperature data accurately and generally agree with the tracer experiment. Hence, they show that the primary clarifier can be modeled by a relatively simple reactor scheme. Because the modeled total volume corresponds approximately to the real reactor volume, the non-existence of shortcuts or dead zones can be assumed; this is important information that would not be available if modeling was based on experience and intuition only.

Computational time

Although the presented method can easily be run on a modern personal computer with reasonable computational...
time, there is much room for improvement. (i) The structure of the model and parameters are optimized simultaneously. A random mutation to the structure is as likely as a change to a parameter value. Because the ERC sets that contain the parameter values are rather large, there can be a large lag before a particular structure co-occurs with a particular parameter value. This lag is a known weakness of GP; ERCs are seen as ‘the skeleton in the closet of GP methods’ (Evett & Fernandez 1998). However, several strategies to overcome this weakness exist (e.g. Evett & Fernandez 1998). (ii) The solution of the ODE system can be very costly for long time series and systems requiring small step sizes. Although system evaluation prior to solving or early discarding strategies could be prescribed, these methods would directly influence the diversity of the population.

**CONCLUSIONS**

In this paper, grammar-based GP was applied to generate hydraulic reactor models. A special tree encoding that can represent hydraulic reactor models as program trees was introduced.

It was shown that, given influent and effluent measurements from a reactor, GP can evolve a reactor model that reproduces the measured effluent series without additional information on the structure of the solution. We suggested evolving several reactor models in different GP runs and let the modeler choose the most appropriate model after taking additional information into account. This latter step is important because a lack of identifiability, poor data quality and limited computing resources all affect the search and can lead to non-optimal solutions.

Although alternative approaches such as tracer experiments are superior to the approach presented here, they have limited applicability in practice. The presented approach, however, is a cost-effective alternative for extracting additional information on transport phenomena without the need for difficult and costly measuring campaigns and thus allows for more accurate modeling of hydraulic processes in practice, which is key to the successful modeling of WWTP.

**REFERENCES**


