Application of Learning Classifier Systems to Gene Expression Analysis in Synthetic Biology

Changhee Han, Kenji Tsuge, and Hitoshi Iba

Abstract Learning classifier systems (LCS) are algorithms that incorporate genetic algorithms (GA) with reinforcement learning (RL) to produce adaptive systems described by simple if-then rules. As a relatively new interdisciplinary branch of biology, synthetic biology pursues the design and construction of complex artificial biological systems from the bottom-up. There is a rising trend in designing artificial metabolic pathways that show previously undescribed reactions produced by the assembly of enzymes from different sources in a single host. However, few researchers have succeeded thus far because of the difficulty of analyzing gene expression accurately. To tackle this problem, data mining and knowledge discovery are essential. In this context, nature-inspired LCS are well suited to extracting knowledge from complex systems, and thus can be exploited to investigate and utilize natural biological phenomena. This chapter focuses on applying LCS to gene expression analysis in synthetic biology. Specifically, we describe the optimization of artificial operon structure for the biosynthesis of metabolic pathway products in *Escherichia coli*. This optimization is achieved by manipulating the order of multiple genes within the artificial operons.

Changhee Han
Graduate School of Information Science and Technology, University of Tokyo, 122B1 Engineering Building 2, 7-3-1 Hongo Bunkyo-ku, Tokyo, Japan, e-mail: han@iba.t.u-tokyo.ac.jp

Kenji Tsuge
Institute for Advanced Biosciences, Keio University, 403-1 Nipponkoku, Daihoji, Tsuruoka, Yamagata, Japan, e-mail: ktsuge@ttck.keio.ac.jp

Hitoshi Iba
Graduate School of Information Science and Technology, University of Tokyo, 101D4 Engineering Building 2, 7-3-1 Hongo Bunkyo-ku, Tokyo, Japan, e-mail: iba@iba.t.u-tokyo.ac.jp

1 Introduction

Nature-inspired learning classifier systems (LCS) [28] were originally envisioned as *cognitive systems* with interactive components, permitting the classification of *massive and noisy datasets* in biological systems. Furthermore, LCS efficiently generate compact *rules describing complex systems* that enable knowledge extraction. Hence, LCS perfectly complement synthetic biology [6], which pursues the design and construction of *complex artificial biological systems*.

In *synthetic biology*, optimizing *gene expression* remains a challenging and potentially fruitful goal that would facilitate the mass production of biofuels, biomedicine, and engineered genomes, among other products. LCS have performed well within many different biological domains, including gene expression analysis [2, 24, 68]. Implementation of LCS to deal with gene expression is poised to grow in the near future, as a number of sizeable genomic datasets are made available by large-scale international projects, such as the 1000 Genomes Project, the 100,000 Genomes Project, and the ENCODE Project.

This chapter presents a crucial first step of applying LCS to complex tasks in synthetic biology, especially in the context of gene expression analysis. This chapter consists of three parts. In the first part, we introduce the framework of LCS and explain how common LCS work. LCS can be used to implement precise data mining methods with an *if-then rule structure* by adopting the strong points of both genetic algorithms (GA) and reinforcement learning (RL). Two major types of LCS exist: Michigan-style LCS, which assess the fitness of individual classifiers, and Pittsburgh-style LCS, which assess the fitness of whole populations. Whereas Pittsburgh-style LCS, such as GAssist and BioHEL, achieve good results in bioinformatics tasks because of their simple rules, Michigan-style LCS are more widely applicable to other practical tasks. For example, minimal classifier systems (MCS) [13] can serve as archetypes for more complex implementations, strength-based zeroth-level classifier systems (ZCS) [62] can solve simple problems precisely, and accuracy-based extended classifier systems (XCS) [63] show optimal solutions within many fields.

The second part of this chapter describes synthetic biology and its potential contributions to society [17]. We also describe how LCS have tackled problems related to gene expression analysis [24] and how LCS can optimize an artificial operon structure. To fulfil the goals of synthetic biology, it is essential to optimize each stage in the *design cycle*, utilize *basic biological blocks*, and assemble *DNA sequences*. LCS can be applied to fields ranging from renewable biofuels to biomedicine—for intractable diseases such as cancer, infectious diseases, and autoimmune disorders—and engineered genomes of artificial organisms. LCS have outperformed gene expression analysis in many instances, especially in cancer classification. *Operons* are functioning units of genomic DNA that are transcribed as a unit and regulated together, thereby coordinating gene transcription. We can maximize the functionality of bacteria, such as *Escherichia coli* (*E. coli*), by modifying operon structure and thus altering gene expression.

The third part of this chapter focuses on our *first computational approach* to analyzing the rules describing the relationship between gene order within an operon and *E. coli* growth (i.e., production) by consultation algorithms, including LCS. We extracted operon construction rules by machine learning and verified those rules by conducting wet-lab experiments on newly designed operons. We identified gene orders that significantly control the growth of *E. coli* and enable the design of new *E. coli* strains with high growth rates. These results indicate that LCS—followed by consultation with well-known algorithms—can efficiently extract knowledge from big and noisy biological system datasets as well as previously intractable dynamic systems.

2 Learning Classifier Systems: Creating Rules that Describe Systems

LCS are nature-inspired machine learning methods; accordingly, they can function as cognitive systems featuring complex interactive components with nonlinear collective activity [28, 29, 31]. For knowledge extraction, LCS generate new random if-then rules by GA and choose the best rule by RL. The domain hierarchy of LCS is described in Fig. 1. To give a general introduction to LCS, this section highlights the basic LCS mechanisms and the fundamental distinction between Michigan- and Pittsburgh-style LCS. More details about implementation optimization as well as a cohesive encapsulation of implementation alternatives are available in an outstanding LCS survey [58]. For a historical overview of LCS research, see [14].

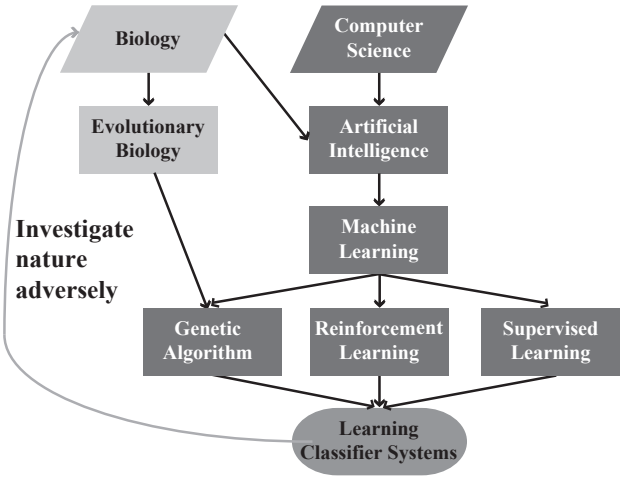

Fig. 1 Domain hierarchy of the LCS concept. Because of their rule comprehensibility, nature-inspired LCS are effective methods to investigate and utilize natural biological phenomena

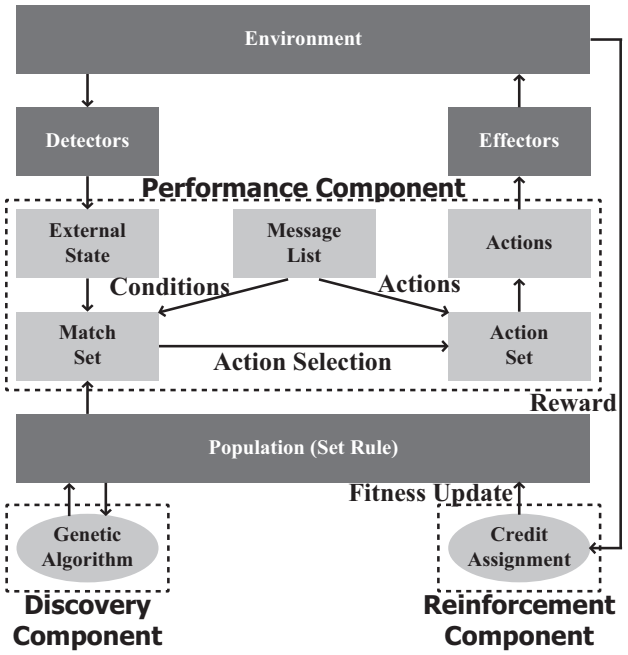

Fig. 2 Interactions among LCS components

2.1 Basic Components

It is not absolutely clear which components should be regarded as basic components of LCS because many different implementations of LCS exist. Holmes et al. [32] outline four practically minimal components: (1) a *finite population of classifiers* that describes the current knowledge of the system to solve distributed problems; (2) a *performance component*, which controls interactions with the environment; (3) a *reinforcement component* (or a credit assignment component [30]), which distributes the reward from the environment to the classifiers to obtain globally optimal solutions; and (4) a *discovery component*, which exploits GA to discover better rules and improve existing ones according to fitness estimates. Fig. 2 demonstrates the highly interactive mechanisms and how these major components interact with each other.

2.2 Michigan- and Pittsburgh-style LCS

LCS research is fundamentally divided into two groups: Michigan-style LCS and Pittsburgh-style LCS. Holland [31], the father of LCS, proposed Michigan-style

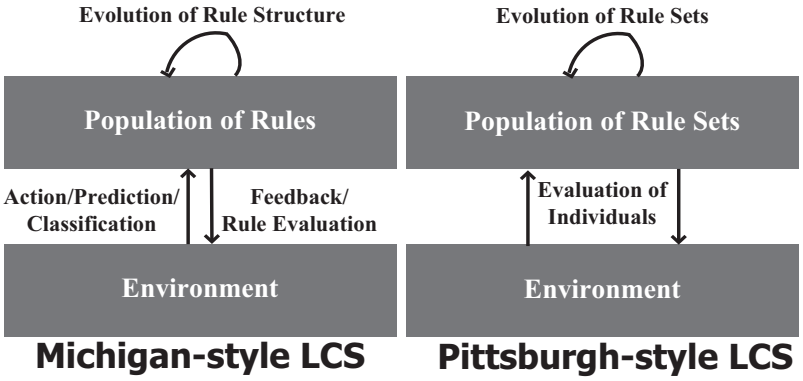

Fig. 3 Differences between Michigan- and Pittsburgh-style LCS. While Michigan-style LCS evolve at the level of individual rules, Pittsburgh-style LCS evolve at the level of multiple rule sets

LCS originally, and Smith [54, 55] later introduced Pittsburgh-style LCS. While in Michigan-style LCS, individual classifiers cooperate to provide a collective solution for the entire population, in Pittsburgh-style LCS, individual complete and variable-length sets of classifiers compete to solve the problem. Fig. 3 illustrates the main structural differences between these two systems.

In general, Michigan-style LCS are more commonly used because of their simplicity, fewer evaluation periods, and higher flexibility in dealing with a wide range of problems. Michigan-style LCS evaluate the fitness of individual classifiers, whereas Pittsburgh-style LCS assess the fitness of entire populations. Therefore, Michigan-style LCS are typically online learning systems that learn iteratively from sets of problem instances, and Pittsburgh-style LCS are more suitable for offline learning systems that learn iteratively from single problem instances. Moreover, Michigan-style LCS show more distributed solutions with a huge number of rules, while Pittsburgh-style LCS provide more compact solutions with few rules. Two examples of Pittsburgh-style LCS implementations are GAssist [7] and its successor BioHEL [8, 21], which is used widely for data mining, especially with large-scale bioinformatic datasets.

3 Examples of LCS

This section describes some widely used Michigan-style LCS.

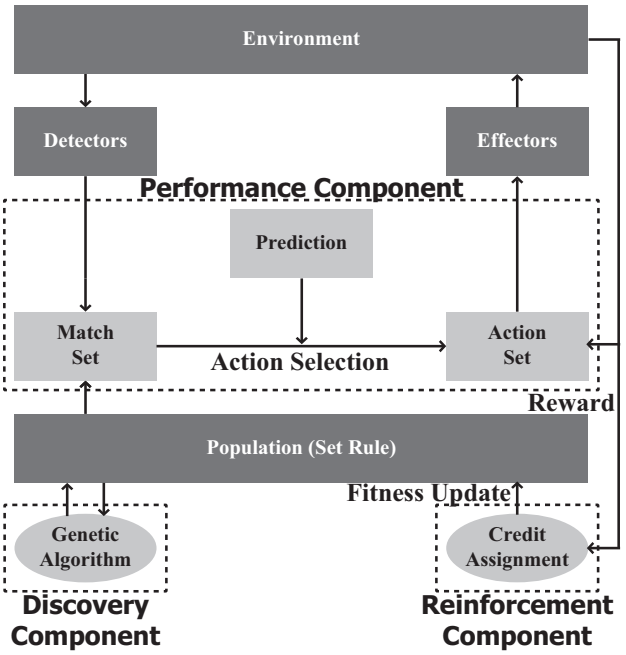

Fig. 4 Interactions among minimal classifier system components

3.1 Minimal Classifier Systems

MCS are simplified LCS implementations that were proposed by Bull [13] as platforms for advancing LCS theory. They provide helpful archetypes for more complicated implementations. Fig. 4 illustrates how MCS work as a whole. Whereas typical LCS possess a message list in order to handle multiple messages both from the environment and other components' feedback at previous time steps, MCS only learn iteratively from one data instance at a time. If the match set conflicts with a given input, a new rule—obtained randomly by GA to handle the input—replaces a previous one. Therefore, a population remains constant according to its fitness value. New rules usually inherit fitness from the previous rules.

The following equation updates the action set [A] at a learning rate β:

$$fitness([A]) \leftarrow fitness([A]) + \beta \left(\left(\frac{Reward}{|[A]|} \right) - fitness([A]) \right)$$

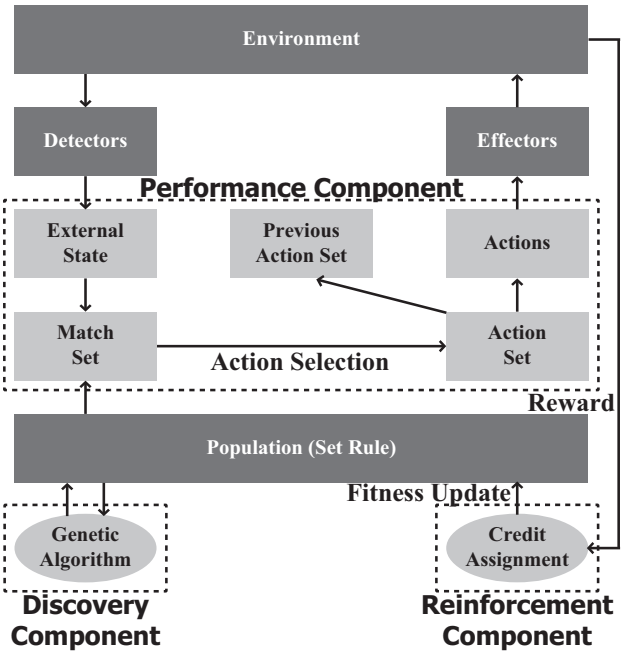

Fig. 5 Interactions among zeroth-level classifier system components

3.2 Zeroth-level Classifier Systems

To increase the clarity and performance of LCS, Wilson [62] proposed simple ZCS for practical use. Fig. 5 shows the interactions among ZCS components. ZCS are often called strength-based LCS (or payoff-based LCS) since their rule fitness depends on the strength (i.e., payoff) defined by the rules. ZCS differ from the typical LCS in that they eliminate the message list and rule-binding, so inputs are straightforwardly matched with the population of rules and no prediction of each action exists. Furthermore, in ZCS, the credit assignment schema basically adopts an implicit bucket brigade [25], which distributes credit between all rules proposing a given action. ZCS perform well in a simple framework relative to original LCS. However, ZCS still find considerably suboptimal solutions in many applications because their simplicity rapidly increases the number of over-general classifiers [18].

The following equation updates the action set [A], defining [A]' as the action set that follows [A] in time, β as a learning rate, and γ as a pre-determined discount factor:

$$\text{fitness}([\text{A}]) \leftarrow \text{fitness}([\text{A}]) + \beta(\text{Reward} + \gamma\,\text{fitness}([\text{A}]') - \text{fitness}([\text{A}]))$$

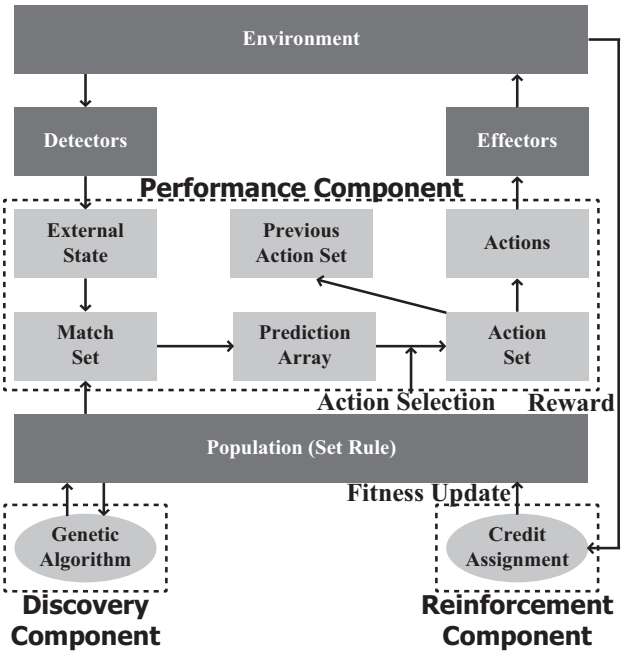

Fig. 6 Interactions among extended classifier system components

3.3 Extended Classifier Systems

One year after introducing ZCS, Wilson [63] proposed XCS to address the drawbacks of ZCS, using accurate and general classifiers. As Fig. 6 illustrates, XCS inherited many ZCS attributes. While ZCS are referred to as strength-based LCS, XCS are called accuracy-based LCS because their rule fitness does not depend on the payoff itself but on the accuracy of payoff predictions [37]. To prevent XCS rules from consistently executing bad actions owing to accuracy-based fitness and to find an optimal action-selection solution, XCS exploit Q-learning [59]—which is a well-known and traditional RL technique that performs well against complex problems—while previous LCS often use TD(0) [56] instead. In addition to using accuracy-based fitness, ZCS replaces standard GA with niche-based GA to eliminate undesirable competition. This combination of accuracy-based fitness, which promotes prediction accuracy, and the niche GA, which accelerates the process of obtaining general rules, makes excellent predictions possible. As a consequence, many researchers have adopted XCS extensively because of their simplicity in generating optimal solutions across a wide range of research fields [1, 47].

The following five steps update the action set [A] with a relative accuracy [15]:

1. Each rule's error ε_j is updated: $\varepsilon_j = \varepsilon_j + \beta(|P - p_j| - \varepsilon_j)$, where β is a learning rate.

2. Rule predictions p_j are updated: $p_j = p_j + \beta(P - p_j)$.
3. Each rule's accuracy κ_j is determined: $\kappa_j = \alpha\left(\frac{\varepsilon_0}{\varepsilon}\right)^{\nu}$ or $\kappa_j = 1$ where $\varepsilon < \varepsilon_0$ and ν, α, and ε_0 control the shape of the accuracy function.
4. A relative accuracy κ'_j is determined by $\frac{\text{each rule's accuracy}}{\text{total accuracy in the action set}}$.
5. The relative accuracy modifies the classifier's fitness F_j: if the fitness is adjusted $1/\beta$ times, $F_j = F_j + \beta(\kappa'_j - F_j)$, otherwise $F_j =$ the average of κ'_j values.

4 Synthetic Biology: Designing Biological Systems

Engineering microorganisms accurately requires a comprehensive understanding of natural biological systems, such as cryptic cellular behaviors, and tools for controlling cells. To overcome these problems, synthetic biology aims to design and construct biological systems for practical applications. In order to avoid empirical studies without undertaking predictive modeling, synthetic biology is shifting from developing *proof-of-concept designs* to establishing *general core platforms* for efficient biological engineering based on *computer science* [17]. Optimizing gene expression computationally plays an essential role in mass-producing useful materials through a synthetic biology approach. This section highlights the key progress and future challenges in synthetic biology and details some potential applications—namely biofuels, biomedicine, and engineered genomes.

4.1 The Synthetic Biology Design Cycle

Fig. 7 shows the design cycle for the core platforms in synthetic biology. To avoid laborious trial and error, this design cycle illustrates how to (1) design systems according to high-level concepts, (2) model these designs as circuits with efficient parts libraries, (3) simulate their functionality, (4) construct the design effortlessly, (5) probe the resulting circuits, and (6) measure the results. In the design cycle, constant feedback between stages plays a key role in enhancing circuit functionality. Moreover, evolutionary strategies exist in the cycle to increase the performance of other steps, though these strategies remain underutilized.

Despite the progress in this field, the design cycle for core platforms remains slow, costly, and arduous; the goals of synthetic biology—understanding and manipulating biological systems—are still exceptionally challenging, as biological phenomena are complex and opaque. Though much progress has been made in several of these steps, others require more innovation. Modeling complex biological systems, in particular, requires novel computational analysis methods.

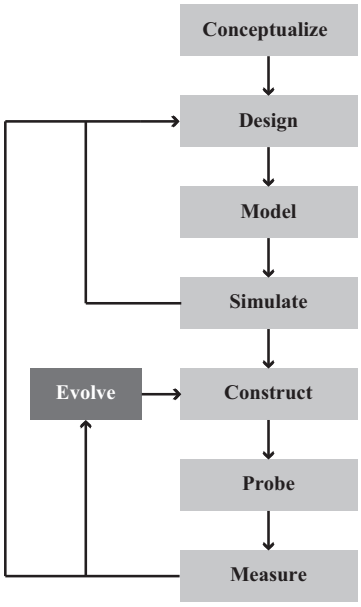

Fig. 7 Design cycle for engineering circuits in synthetic biology. Phases such as circuit conceptualization, design, and construction have advanced significantly, but many bottlenecks still exist at modeling, simulation, probing, and measurement phases

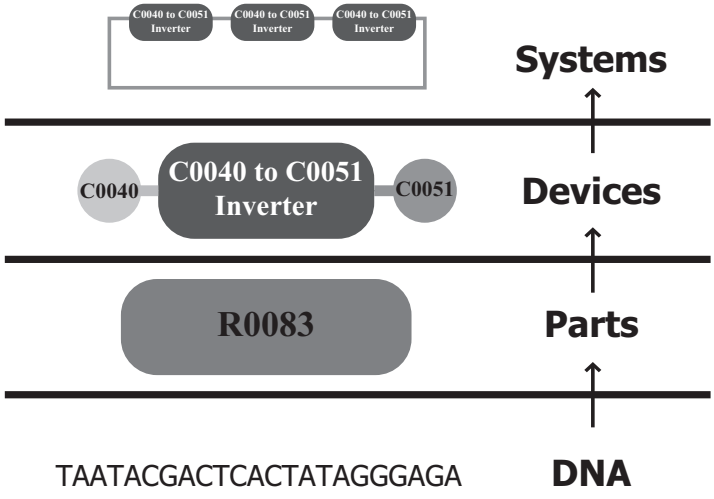

Fig. 8 Hierarchy of synthetic biology. DNA sequences form parts, parts form devices, and devices form systems

4.2 Basic Biological Parts

In synthetic biology, basic universal parts are commonly used to build higher-order synthetic circuits and pathways in order to simplify biological engineering (Fig. 8).

These components fulfil particular functions, such as controlling transcription, regulating molecules, and changing genetic material. Kushwaha and Salis [38] established regulatory genetic parts that can transfer a multienzyme circuit and pathway between *Bacillus subtilis*, *Pseudomonas putida*, and *E. coli*. These basic parts usually come from natural systems. For example, we exploit natural promoters, such as P_L of phage lambda and the *lac* operon promoter, to regulate gene expression, both of which occur naturally in prokaryotes [20, 67].

4.3 DNA Construction

As synthetic biology designs complex biological systems from the bottom-up, standardized biological parts are vital. Parts can form devices, devices can be organized into systems, and eventually systems can create large networks, synthetic chromosomes, and genomes. To accomplish this goal, we must assemble DNA for the functional circuits.

Conventionally, the assembly of DNA sequences required laborious cloning steps, so constructing complex metabolic pathways and regulatory circuits was impossible; to deal with DNA sequences effortlessly, sequence-independent, one-pot, and ordered assembly methods have been recently explored. As a result, many assembly methods have been proposed, such as the BioBrick and BglBrick standards [5], sequence- and ligase-independent cloning (SLIC) [41], and Gibson assembly [23]. Advancements in DNA synthesis technologies may lead to direct circuit design without multiple parts in the future, though this is currently impractical owing to the difficulty of constructing large combinatorial libraries of artificial circuits.

4.4 Future Applications

Standardized biological parts and circuit design can realize the goals of synthetic biology—designing and constructing biological systems for useful purposes.

4.4.1 Biofuels

Fossil fuels account for more than 80% of the world's energy supply, even though they harm the environment significantly and are nonrenewable. They must be replaced with renewable substitutes in the near future. Biofuels made from renewable and eco-friendly resources may become a great alternative, if they can compete economically.

Synthetic biology has pursued engineering many microbial hosts to produce various biofuels, including long-chain alcohols, fatty acids, alkanes, and alkenes [52].

To produce biofuels, it is crucial to characterize the enzymes, regulation, and thermodynamics of relevant native metabolic pathways. Until now, the most efficient production has been achieved with *E. coli* because of their significant resiliency, sufficient toolbox, and peripheral metabolic pathways. Some advanced biofuel production methods have already achieved industrial level production, though most are far from it.

4.4.2 Biomedicine

Functional devices—made by standardized biological blocks—can provide novel diagnostic and therapeutic tools for previously incurable or intractable diseases, such as cancer, infectious diseases, and autoimmune disorders. To achieve this goal, it is necessary to identify the safety, side effects, and *in vivo* performance of engineered circuits. In the context of therapy, it is especially important to determine how to deliver engineered cells and synthetic gene constructs to internal tissues.

Cancer

One factor that impedes curing cancer is the difficulty of differentiating between cancerous and healthy cells. The signature of hypoxia can help differentiate those cells. Therefore, synthetic biologists have engineered *E. coli* to attack mammalian cells selectively in hypoxic environments using heterologous sensors [4]. Other researchers have followed a similar logic to take advantage of the link between enzymatic activity and hypoxia [64].

Furthermore, Xie et al. [65] proposed a miRNA-detection strategy that can distinguish between cell types according to miRNA expression signatures. Separate circuits can identify a cell type by miRNA expression level; the combination of these circuits can perform as a cell-type classifier according to miRNA level. This approach has been shown to selectively kill HeLa cancer cells by regulating the expression of a proapoptosis protein by two high-expression and three low-expression miRNAs. In this way, different circuits can be blended to increase diagnosis and therapy performance.

Infectious diseases

Antibiotic resistance and other properties, such as biofilm formation [44] and persistence [3], have intensified microbial infections. Treating severe antibiotic resistance in pathogens often requires the use of combinations of powerful antibiotics, which may ultimately promote further antibiotic resistance. Moreover, antibiotics can cause dreadful disorders of the human microbiome because they kill both pathogenic and non-pathogenic bacteria. As in the case of cancer, selective targeting of pathogens is essential in order to minimize this side effect. Other approaches are

also possible through synthetic biology, such as targeting bacterial biofilms, reinforcing antibiotics [45], and engineering new treatment vehicles.

Autoimmune disorders

Autoimmune disorders occur when the human immune system errantly attacks healthy body tissue covered in autoantigens. Autoimmune diseases include Hashimoto's thyroiditis, reactive arthritis, and type I diabetes. Since precise causes are unknown, research programs should search for all potential targets. Larman et al. [39] utilized a synthetic human peptidome to find potential autoantigens and antibodies. This method enabled the investigation of all the coding regions within the human genome.

4.4.3 Engineered Genomes

Synthetic biology also aims to create artificial organisms and reveal the complexity of genetic variation with new enzymatic tools that enable precise genetic modifications. Recent advances in genetic assembly tools for synthetic biology have enabled directed mutagenesis to identify gene function, the correction of defective genes, and the redesign genome structure. For example, Elowitz et al. [19] engineered genomes in an attempt to build a new organism and thus discover principles underlying life. In *E. coli*, Isaacs et al. [34] manipulated chromosomes by replacing all 314 TAG stop codons with TAA codons.

5 Gene Expression Analysis with LCS

Because LCS originated from biological systems and contain complex, interactive components that allow them to function as cognitive systems, they are well suited for classifying large-scale, noisy datasets that are often encountered in synthetic biology and systems biology. Furthermore, LCS generate sufficient rules describing systems with only a few interpretable key attributes, whereas state-of-the-art machine learning methods such as deep learning [26] cannot determine rules; LCS perform excellently for knowledge extraction to understand complex biological phenomena. Accordingly, LCS and LCS-inspired systems have solved many bioinformatic and medical problems, such as automating alphabet reduction for protein datasets [9] and the classification of a primary breast cancer dataset [36]. LCS-inspired systems, like BioHEL [8, 21], exist that are even designed for data mining large-scale bioinformatic datasets.

As explained in the previous section, gene expression analysis plays an essential role in solving a large number of problems in synthetic biology. Machine learning techniques have recently been attempted in a wide range of genetics and genomics

fields related to the mechanisms of gene expression because of their ability to interpret large and complex genomic datasets [42]. For instance, machine learning can predict (1) gene expression from DNA sequences [10], (2) ChIP-seq profiles of histone modifications [35], and (3) transcription factors that bind to a gene promoter [49]. Researchers have also attempted to model all gene expression in a cell using a network model [22]. This trend will accelerate dramatically with the rapid advance of machine learning algorithms, tremendous increases in computational power, and accessibility of massive datasets from large-scale international collaborations, such as the 1000 Genomes Project, the 100,000 Genomes Project, and the ENCODE Project.

Glaab et al. [24] evaluated rule-based evolutionary machine learning systems inspired by Pittsburgh-style LCS, specifically BioHEL, and GAssist, in order to increase the understandability of prediction models with high accuracy. Using three publicly available microarray cancer datasets, they inferred simple rule-based models that achieved an accuracy comparable with state-of-the-art methods, such as support vector machines, random forest, and microarray prediction analysis. Furthermore, essential genes contained within BioHEL's rule sets beat gene rankings from a traditional ensemble feature selection. Specifically, BioHel better predicted the relationships between relevant disease terms and top-ranked genes.

Zibakhsh et al. [68] suggested memetic algorithms, including LCS with a multi-view fitness function approach. Fitness functions have two evaluation procedures: the evaluation of each individual fuzzy if-then rule in accord with the specified rule quality and the evaluation of each fuzzy rule quality according to the whole rule set performance. These algorithms significantly outperformed classic memetic algorithms in discovering rules. These new approaches have also extracted understandable rules differentiating different types of cancer tissues that were more accurate than other machine learning algorithms, such as C4.5, random forest, and logistic regression model.

Abedini et al. [2] proposed two XCS-inspired evolutionary machine learning systems to investigate how to improve classification accuracy using feature quality information: FS-XCS, which utilize feature selection to reduce features, and GRD-XCS, which exploit feature ranking to modify the rule discovery process of XCS. The authors classified breast cancers, colon cancers, and prostate cancers, each with thousands of features, from microarray gene expression data. They found that feature quality information can help the learning process and improve classification accuracy. However, approaches that restrict the learning process and rely exclusively on the feature quality information, such as feature reduction, may decrease classification accuracy.

6 Optimization of Artificial Operon Structure

A number of genes in genomes encode many proteins that modulate cellular activities or implement specific functionality. In bacterial genomes, such as *E. coli*

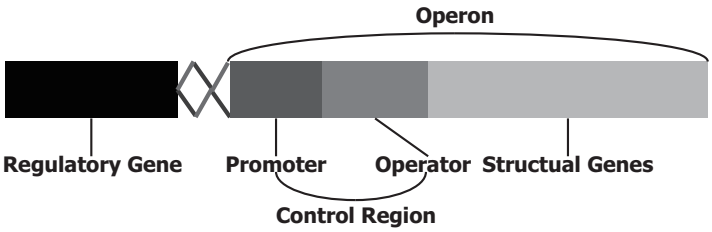

Fig. 9 Operon structure consisting of a promoter, an operator, and structural genes. The operon is controlled by a regulatory gene

genomes, such genes frequently act as an operon, that is, a functioning unit of genomic DNA that controls the transcription of multiple genes simultaneously with a single promoter. Fig. 9 illustrates a typical operon with a promoter, an operator, and structural genes. An operon is transcribed into a continuous mRNA strand and either translated in the cytoplasm, or trans-spliced to generate monocistronic mRNAs that are translated independently. As such, gene expression of elements within an operon decrease linearly with transcription distance [43]. To increase the productivity of synthetic metabolic pathways, the relative abundance of transcripts must be regulated accurately using an operon to achieve the balanced expression of multiple genes and avoid the accumulation of toxic intermediates or bottlenecks that inhibit the growth of microorganisms [61].

To approach this central challenge in synthetic biology, operon structure optimization has been pursued in recent years. There are several examples: the optimization and genetic implementation of a blueprint as an artificial operon, according to metabolic real-time analysis, tripled the production of dihydroxyacetone phosphate from glucose [12]; the amplification of genomic segments in artificial operons successfully controlled gene expression using selective RNA processing and stabilization (SRPS) [53], which transcribes primary mRNA into segments using nucleases and thus produces variation in stability among the segments [66]; libraries of tunable intergenic regions (TIGRs)—which recombine numerous post-transcriptional control elements and permit specifying the desired relative expression levels—have helped optimize gene expression in artificial operons [50].

Instead of modifying genes themselves, a completely different approach—reordering multiple genes into an operon structure with an appropriate promoter—may become a breakthrough in optimizing *E. coli* growth (i.e., production). A novel gene assembly method, the ordered gene assembly in *Bacillus subtilis* (OGAB) method [57], enables the changing of gene orders by the assembly of multiple DNA fragments in a fixed order and orientation. Newly designed operons, created by reordering genes from metabolic pathways, show significant differences in production because the abundance of specific mRNA sequences decreases as genes become separated from the promoter in *E. coli*.

The production of zeaxanthin increased considerably by rearranging the gene order of a five-gene operon [46, 48]. Similar results were also observed in the pro-

duction of UHMW-P(3HB), which was achieved by reordering gene order within a three-gene operon [27].

However, optimizing more than five genes remains laborious as the number of gene orders increases factorially—a five-gene operon possesses 120 gene orders, but a ten-gene operon possesses 3,628,800 gene orders. Moreover, this research is typically carried out empirically without predictive modeling. As a result, a computational approach that optimizes operons accurately with respect to gene order is essential for comprehensive prediction. The next section presents our novel computational approach to analyzing the relationship between gene order within an operon and the growth rate of *E. coli* using a machine learning algorithm, such as LCS. We not only simulated those operon construction rules, but also verified the rules by conducting wet-lab experiments with newly designed operons.

7 Optimization of Artificial Operon Construction by Machine Learning

7.1 Introduction

Our aim was to investigate the influence of gene order among ten genes within an operon on the growth rate of *E. coli*. To do so, we utilized consultation via machine learning algorithms that include LCS to analyze the relationship and then verified predicted gene orders with high growth rates using wet-lab experiments. This is the first computational approach for analyzing the relationship between operon gene order and *E. coli* growth rate. This research significantly contributes to the goal of designing efficient operons for the mass-production of useful materials in synthetic biology—such as personalized medicine [60], medical diagnosis [16], and advanced biofuels [40]. Furthermore, this study provides a crucial step toward interdisciplinary research linking LCS and synthetic biology, which can be applied to various other tasks.

7.2 Artificial Operon Model

We applied machine learning to investigate the construction principles relating gene order within an operon to the separation from a promoter involved in a metabolic pathway and thus the growth rate of *E. coli*. The operon contained ten genes (labeled A, B, C, D, E, F, G, H, I, and J). We then verified the high-growth-rate gene orders using wet-lab experiments.

The expression of multiple genes changes in accordance with changes in gene order [61]. Therefore, *E. coli* growth rates differ not only as a result of the presence and absence of genes, but also gene order within operons. Generally, if the expres-

sion of multiple genes is balanced, *E. coli* grows rapidly, but if it is not, *E. coli* grows poorly or is totally inhibited as toxic intermediates or bottlenecks accumulate. Yet, optimizing more than five genes has been impossible using conventional approaches because the number of gene orders increases factorially with the number of genes in an operon.

We identified ten genes in this study. It was challenging to analyze them accurately for the following reasons: (1) the number of gene orders obtained from wet-lab experiments was only 0.004% of the total 3,628,800 gene orders (the dataset consists of 93 individual datasets from *E. coli* with gene orders similar to wild-type strains as well as 51 datasets with random gene orders); (2) even *E. coli* with identical gene orders exhibit a large standard deviation in growth rate (the maximum standard deviation of our dataset is around 0.05/h); (3) *E. coli* strains with high growth rates in the dataset possess similar gene orders—the highest growth rate was approximately 0.73/h. Therefore, we adopted algorithms of heuristic machine learning techniques that can resolve the trade-off between accuracy and smoothness to elucidate which gene orders significantly influence *E. coli* growth. As Fig. 10 shows, gene order refers to the arrangement of genes, and it significantly influences

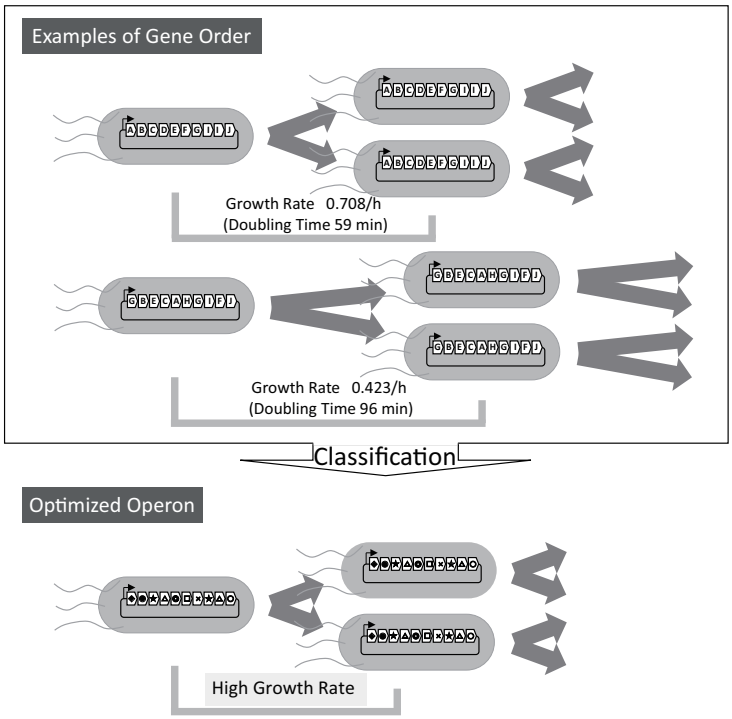

Fig. 10 Optimizing gene order of an artificial operon. The growth rate of *E. coli* differs with gene order, so we investigated operon construction principles relating gene order to growth rate and tried to design new *E. coli* strains with high growth rates

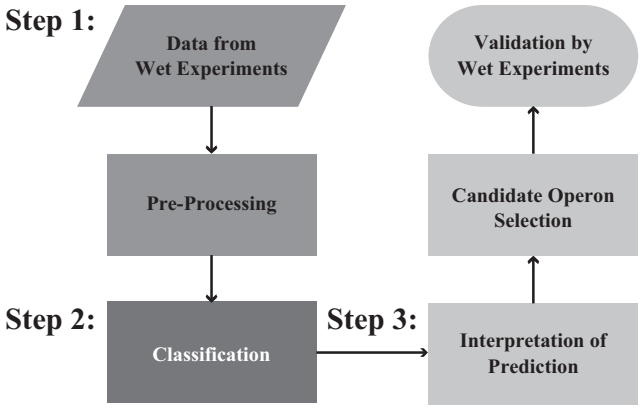

Fig. 11 Flowchart illustrating the experimental procedure. The protocol consists of three steps: (1) pre-processing; (2) supervised analysis; and (3) post-analysis

the growth rate of *E. coli*. Optimization can improve the efficiency of metabolic biosynthesis.

(Fig. 11) provides a flowchart describing the experimental procedure. Throughout the wet-lab experiment, *E. coli* was cultivated in duplicate. The OGAB method was exploited to reconstitute gene orders by assembling multiple DNA fragments with a desired order and orientation and thus generating an operon structure in a resultant plasmid. We normalized the raw data and classified the gene orders into several growth rates; we selected a sampling of gene orders and verified them through wet-lab experiments.

7.3 Experimental Framework

We conducted two experiments with consultation algorithms, one employing the results of two algorithms—Pittsburgh-style LCS [28, 29, 31] called GAssist [7] for compact solutions with few rules and a decision-tree induction technique, C5.0—and the other employing the results of four algorithms—other two well-known conventional machine learning techniques, random forest and multilayer perceptron, in addition to using LCS and C5.0. This was executed in order to analyze the relationship between the gene order within operons and *E. coli* growth rate. Then, we designed six operons per experiment according to the optimal predicted gene orders. We use the term "consultation" to refer to choosing gene orders with high growth rates in each algorithm by considering attributes of classifiers from C5.0 [51] in order to avoid over-fitting. The problem becomes a classification domain, and the parameters of the classifier model were selected to maximize ten-fold cross-validation accuracy.

Table 1 Classes of growth rates. Growth rate is classified into one of eight classes defined by an equal interval

Classes	Growth Rate	Classes	Growth Rate	Classes	Growth Rate
0	0–0.1/h	0.1	0.1–0.2/h	0.2	0.2–0.3/h
0.3	0.3–0.4/h	0.4	0.4–0.5/h	0.5	0.5–0.6/h
0.6	0.6–0.7/h	0.7	$\geq$ 0.7/h		

Table 2 Classes of growth rates. Each growth rate is divided into seven classes with smaller ranges for high growth rates in order to predict gene orders with high growth rates more accurately

Classes	Growth Rate	Classes	Growth Rate	Classes	Growth Rate
0	0–0.4/h	0.4	0.4–0.5/h	0.5	0.5–0.6/h
0.6	0.6–0.65/h	0.65	0.65–0.7/h	0.7	0.7–0.72/h
0.72	$\geq$ 0.72/h				

7.3.1 Consultation with Two Algorithms

Using LCS and C5.0, we classified 45 explanatory variables (describing the relative orders between two genes) into eight growth rate groups (Table 1)—the growth rates take precise values, so they had to be converted into finite classes for the classification task. To test the classification performance in wet-lab experiments, we also examined 20 random datasets out of the total 144 datasets as a test dataset.

Based on these classification results, we selected six gene orders within the operon that are predicted to increase *E. coli* growth, and we then designed strains in order to test them using wet-lab experiments. First, we identified four gene orders that were classified as promoting growth rates exceeding 0.7/h in every algorithm considering C5.0 attributes. Furthermore, we selected two gene orders predicted to have growth rates that are relatively high but significantly different from those of the original gene expression dataset to investigate the influence of modifying gene order.

7.3.2 Consultation with Four Algorithms

In addition to using LCS and C5.0, we also exploited random forest [11], which is an efficient ensemble learning method that employs many decision trees, and multilayer perceptron [33], which is a standard neural network model. We classified 45 explanatory variables (again, describing the relative orders between two genes) into seven growth rate groups (Table 2). The results of the previous experiment (six datasets) were also used as the main dataset for this analysis and 21 additional datasets from the total 150 datasets were used as the test dataset; three datasets per class were employed as test data.

From these classification results, we selected six gene orders for the operon structure and designed them for subsequent wet-lab experiments. First, we identified two gene orders that were estimated to promote growth rates in excess of 0.72/h by our LCS analysis and 0.7/h in the C5.0 analysis and random forest analysis, which considers the C5.0 attributes. In addition, we selected two gene orders that were estimated to promote growth rates in excess of 0.7/h by our LCS, C5.0, and random forest analyses and in excess of 0.65/h by our multilayer perceptron analysis, which considers C5.0 attributes. Finally, two gene orders were identified that were classified as promoting growth rates in excess of 0.7/h by our LCS and random forest analyses, 0.65 by our C5.0 analysis, and 0.72 by our multilayer perceptron analysis, which considers C5.0 attributes.

7.4 Results

This subsection shows how our consultation algorithms utilizing LCS work in cases of consultation using two and four algorithms. The results include both computational simulations and their biological verification. Our method performed effective data mining in this gene expression analysis owing to the high accuracy of LCS and its ability to determine definite understandable rules that describe complex systems efficiently.

7.4.1 Consultation with Two Algorithms

Classification of Test Data by LCS

Classification succeeded with 25% accuracy (Fig. 12); if up to one class error is allowed, we achieved 80% classification accuracy. Most gene orders were classified as promoting growth rates between 0.4 and 0.5.

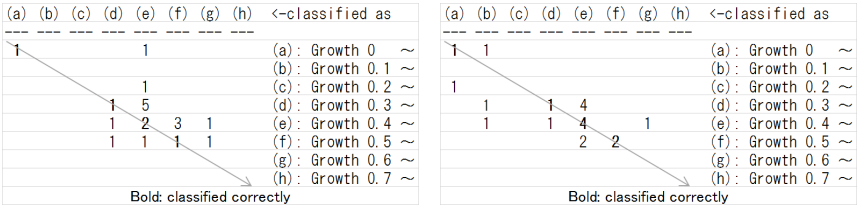

Fig. 12 (a) Classification of test data by learning classifier systems (b) Classification of test data by C5.0. The numbers of gene orders are presented with both the classified class and the actual class

According to a rule set of eight simple and interpretable rules, a gene order was determined that was given the highest class assignment ($\geq$ 0.7/h; the $\rightarrow$ operator represents the rule stating that the gene preceding the operator is located in front of the gene following the operator; for example, A$\rightarrow$B means gene A is located before gene B). According to this rule set, gene A tends to be assigned to the front of the operon, while gene J tends to be assigned to the back. The rule set inferred to describe the highest growth rate classification using LCS is as follows.

- A$\rightarrow$B, A$\rightarrow$G, B$\rightarrow$H, C$\rightarrow$I, D$\rightarrow$F, E$\rightarrow$I, E$\rightarrow$J, H$\rightarrow$J

Classification of Test Data by C5.0

C5.0 produced classifications with 40% accuracy (Fig. 12); permitting one error class, this method obtained 80% of classification accuracy. Six gene orders promoting growth rates between 0.3 and 0.4 or between 0.5 and 0.6 were incorrectly classified as promoting growth rates between 0.4 and 0.5.

Growth Rates of Newly Designed Operons

We designed novel operons based on these classification results, and they exhibited high growth rates (Fig. 13). In particular, Order2 gave a growth rate comparable

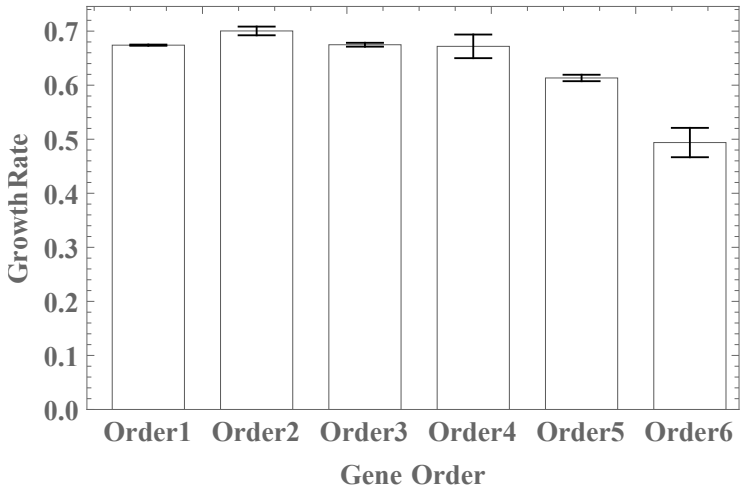

Fig. 13 Growth rates of *E. coli* with novel gene orders shown as means $\pm$ SD. We selected six gene orders within the operon that were classified by our two-algorithm consultation to promote high growth rates and then we designed them for wet-lab verification experiments. We cultured the novel strain in duplicate and estimated the error to be less than 6% per sample. Strains (Gene orders): Order1 (ABEICHDFGJ); Order2 (ABDCEGFIHJ); Order3 (ABCDEIHFGJ); Order4 (ABCDFGEIHJ); Order5 (AGBIEJCHDF); and Order6 (AEICHDFBGJ)

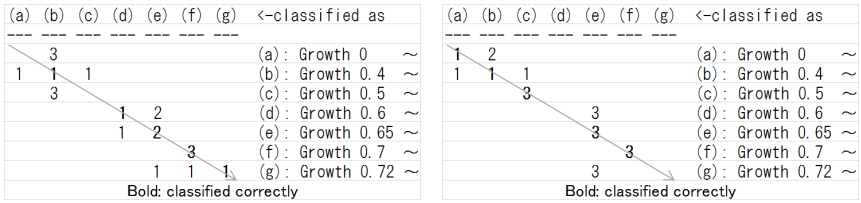

Fig. 14 (a) Classification of test data by learning classifier systems (b) Classification of test data by C5.0. The numbers of gene orders are presented with both the classified class and the actual class

to the highest in the dataset (around 0.73/h) considering that the maximum standard deviation of our dataset is around 0.05/h. Order5 and Order6, which each have completely different gene orders from the dataset, demonstrated low growth rates compared with the other orders.

7.4.2 Consultation with Four Algorithms

Classification of test data by LCS

LCS yielded classifications that succeeded with around 38% accuracy (Fig. 14); if up to one class error is allowed, we achieved around 95% classification accuracy. Six gene orders promoting growth rates between 0 and 0.4 or between 0.5 and 0.6 were classified incorrectly as promoting growth rates between 0.4 and 0.5.

This analysis produced a restrictive rule set of 13 rules that determined a gene order that was assigned to the highest class ($\geq$0.72/h). Genes A, B, C, and D each tended to be assigned to the front of the operon, while gene J tended to be assigned to the back. The rank order of gene A has an especially strong influence on the growth rate. The rule set inferred to describe the highest growth rate classification using LCS is as follows.

- A$\rightarrow$B, A$\rightarrow$C, B$\rightarrow$D, B$\rightarrow$G, C$\rightarrow$G, C$\rightarrow$I, D$\rightarrow$H, E$\rightarrow$I, F$\rightarrow$J, G$\rightarrow$J, H$\rightarrow$E, H$\rightarrow$F, I$\rightarrow$J

Classification of Test Data by C5.0

C5.0 produced a classification with approximately 52% accuracy (Fig. 14); within one class error, we obtained 86% classification accuracy. Six gene orders promoting growth rates between 0.6 and 0.65 or exceeding 0.72 were incorrectly classified as promoting growth rates between 0.65 and 0.7.

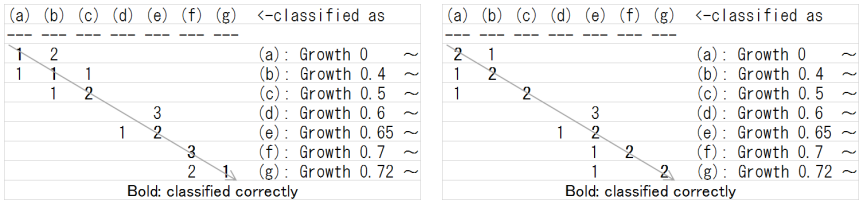

Fig. 15 (a) Classification of test data by random forest (b) Classification of test data by multilayer perceptron. The numbers of gene orders are presented with both the classified class and the actual class

Classification of Test Data by Random Forest

The random forest analysis performed classification with around 48% accuracy (Fig. 15); if up to one class error is allowed, this method reached 100% classification accuracy. Three gene orders promoting growth rates between 0.6 and 0.65 were incorrectly classified as promoting growth rates between 0.65 and 0.7.

Classification of Test Data by Multilayer Perceptron

The multilayer perceptron classification yielded 57% accuracy (Fig. 15); within one class error, this method achieved 90% classification accuracy. Five gene orders promoting growth rates between 0.6 and 0.65 or exceeding 0.7 were incorrectly classified as promoting growth rates between 0.65 and 0.7.

Growth Rates of Newly Designed Operons

As illustrated in Fig. 16, all of the newly designed operons showed high growth rates ($>$0.6/h). However, no operon promoted a higher growth rate than that which was obtained by the previous experiment based off of two-algorithm consultation. Order5, as shown in Fig. 16, demonstrated a large standard deviation (approximately 0.04). The gene orders obtained from the four-algorithm analysis are more similar to each other compared with those obtained in the two-algorithm analysis.

7.5 Conclusion

We found that certain two-gene order rules can be used to design operons that significantly affect *E. coli* growth (i.e., production) by consultation via machine learning algorithms that include LCS. Moreover, we also successfully created new *E. coli*

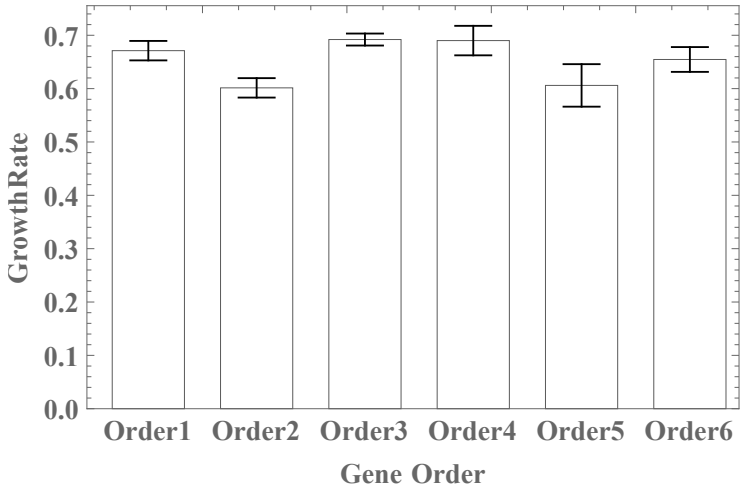

Fig. 16 Growth rates of novel *E. coli* strains presented as means ± SD. We designed six operons with gene orders predicted to have high growth rates according to our four-algorithm consultation. The empirical error was less than 7% per sample. Strains (Gene orders): Order1 (ABCDFEHIGJ); Order2 (ACBDFGEHIJ); Order3 (ABDCEFIGHJ); Order4 (ABDCGEHFIJ); Order5 (ABCDFEIGHJ); and Order6 (ABCDFIEGHJ)

strains with high growth rates using these operon construction rules. Genes that are closer to the promoter in an operon exhibit higher mRNA expression in general, as supported by real-time RT-PCR results. However, the explanation for severe growth rate differences among strains with different novel operons is unclear. The interactions between genes may differ substantially as their sequential order change. However, other potential explanations of the relationship between gene order and growth rate warrant consideration.

Most operon rearrangement studies suggest that the gene orders that resemble those of wild-type strains tend to have high growth rates [46, 48]. Our computationally optimized operons are similar to wild-type strains in gene order to some extent, as expected. Gene A, which exhibits much higher mRNA expression levels than other genes, was consistently located in the front of the operon in order to obtain a high growth rate. However, except for several genes strongly linked to their positions, reordering genes for growth optimization is possible. LCS rule sets, such as the optimal spatial relationship between gene E and gene I, provide further details on how gene orders can be optimized. While this is difficult to predict experimentally, the results are easily understood by biologists.

Our study also reveals that surpassing the highest growth rate of the dataset is challenging for the following reasons: (1) although the classification of test data went well, no operon promoted growth rates exceeding those previously found; (2) the dataset is small and noisy; and (3) the dataset is biased and efficient operons share similar gene orders.

We aimed to optimize operon structure for the largest number of genes thus far, ten genes, which can be rearranged into 3,628,800 orders, by a novel computational approach. We focused on relative orders between two genes as the explanatory variables. However, more comprehensive variables, such as those used in natural language processing, may enhance classification accuracy; the structure involved in assessing sentences through word order is similar to the operon structure involved in predicting growth rates. These findings should also be confirmed using more randomized data to avoid over-fitting. Furthermore, we focused on operons that promote efficient growth; however, future studies should also explore operons that inhibit *E. coli* growth.

Taken together, our findings illustrate that machine learning—especially the use of consultation algorithms utilizing LCS to avoid over-fitting—can help identify the most efficiently structured operons. Changes in mRNA expression of genes and gene interactions altered by gene order may cause these results. Computational results must be interpreted with caution, but newly designed operons tested in wet-lab experiments support this approach. This first computational study proves that pairwise order relationships between genes produce significant differences in operon efficiency; given the difficulty of understanding all interactions between genes, future studies with more comprehensive explanatory variables are needed. Furthermore, our study suggests that LCS can play a significant role in data mining from large and noisy datasets extracted from biological systems, especially gene expression analysis in synthetic biology.

8 Summary

In this chapter, we have described what LCS and synthetic biology are and how they are closely related in the context of gene expression analysis. Specifically, we have illustrated a first computational approach to optimizing operon structure by altering gene order to match optimization rules produced by consultation with machine learning algorithms that use LCS. Our research included both computational simulation and biological verification. Overall, these results indicate that LCS and LCS-inspired systems can perform effective data mining and knowledge discovery in gene expression analysis and synthetic biology broadly because LCS can extract definite understandable rules describing complex systems efficiently while retaining high accuracy.

This chapter both extends LCS and connects LCS with synthetic biology. Our study confirms that LCS—followed by consultation with well-known algorithms to avoid over-fitting and obtain better solutions—can provide excellent knowledge discovery from huge and noisy datasets in biological systems or previously intractable dynamic systems. Moreover, we provided a crucial first step of interdisciplinary research linking LCS and synthetic biology by illustrating both the core concepts as well as a clear relationship between these domains that can be applied to various other tasks.

Acknowledgements The authors thank Dr. Yoshihiko Hasegawa and Dr. Hiroshi Dohi of The University of Tokyo for insightful discussions that were significant in our analysis of the operon structure. We would also like to especially thank UTDS members Marishi Mochida, Akito Misawa, Takahiro Hashimoto, Kazuki Taniyoshi, Yuki Inoue, and Mari Sasaki for making our writing environment so pleasant.

References

1. Abedini, M., Kirley, M.: Guided rule discovery in XCS for high-dimensional classification problems. In: Wang, D., Reynolds, M. (eds.) AI 2011: Advances in artificial intelligence, pp. 1-10. Springer, Heidelberg (2011)
2. Abedini, M., Kirley, M., Chiong, R.: Incorporating feature ranking and evolutionary methods for the classification of high-dimensional DNA microarray gene expression data. Australas. Med. J. (2013) doi: 10.4066/AMJ.2013.1641
3. Allison, K.R., Brynildsen, M.P., Collins, J.J.: Metabolite-enabled eradication of bacterial persisters by aminoglycosides. Nature. (2011) doi: 10.1038/nature10069
4. Anderson, J.C., Clarke, E.J., Arkin, A.P., Voigt, C.A.: Environmentally controlled invasion of cancer cells by engineered bacteria. J. Mol. Biol. (2006) doi: 10.1016/j.jmb.2005.10.076
5. Anderson, J.C., Dueber, J.E., Leguia, M., Wu, G.C., Goler, J.A., et al.: BglBricks: A flexible standard for biological part assembly. J. Biol. Eng. (2010) doi: 10.1186/1754-1611-4-1
6. Andrianantoandro, E., Basu, S., Karig, D.K., Weiss, R.: Synthetic biology: new engineering rules for an emerging discipline. Mol. Syst. Biol. (2006) doi: 10.1038/msb4100073
7. Bacardit, J.: Pittsburgh genetics-based machine learning in the data mining era: representations, generalization, and run-time. PhD thesis, Ramon Llull University, Barcelona (2004)
8. Bacardit, J., Burke, E.K., Krasnogor, N.: Improving the scalability of rule-based evolutionary learning. Memetic Computing. **1**, 55-67 (2009)
9. Bacardit, J., Stout, M., Hirst, J.D., Valencia, A., Smith, R.E., et al.: Automated alphabet reduction for protein datasets. BMC Bioinformatics. (2009) doi: 10.1186/1471-2105-10-6
10. Beer, M.A., Tavazoie, S.: Predicting gene expression from sequence. Cell. **117**, 185-198 (2004)
11. Breiman, L.: Random Forests. Mach. Learn. **45**, 5–32 (2001)
12. Bujara, M., Schmperli, M., Pellaux, R., Heinemann, M., Panke, S.: Optimization of a blueprint for *in vitro* glycolysis by metabolic real-time analysis. Nat. Chem. Biol. **7**, 271–277 (2011)
13. Bull, L.: Two simple learning classifier systems. In: Bull, L., Kovacs, T. (eds.) Foundations of learning classifier systems, pp. 6389. Springer, Heidelberg (2005)
14. Bull, L.: A brief history of learning classifier systems: from CS-1 to XCS and its variants. Evol. Intell. **8**, 55–70 (2015)
15. Bull, L., Bernado-Mansilla, E., Holmes, J.: Learning classifier systems in data mining: An introduction. In: Bull, L., Bernado-Mansilla, E., Holmes, J. (eds.) Learning classifier systems in data mining, pp. 115. Springer, Heidelberg (2008)
16. Chen, Y.Y., Smolke, C.D.: From DNA to targeted therapeutics: bringing synthetic biology to the clinic. Sci. Transl. Med. (2011) DOI: 10.1126/scitranslmed.3002944
17. Cheng, A.A., Lu, T.K.: Synthetic biology: An emerging engineering discipline. Annu Rev Biomed Eng. (2012) doi: 10.1146/annurev-bioeng-071811150118
18. Cliff, D., Ross, S.: Adding temporary memory to ZCS. Adapt. Behav. **3**, 101–150 (1994)
19. Elowitz, M., Lim, W.A.: Build life to understand it. Nature. **468**, 889-890 (2010)
20. Elvin, C.M., Thompson, P.R., Argall, M.E., Hendr, N.P., Stamford, P.J., et al.: Modified bacteriophage lambda promoter vectors for overproduction of proteins in *Escherichia coli*. Gene. **87**, 123-126 (1990)
21. Franco, M.A., Krasnogor, N., Bacardit, J.: Analysing BioHEL using challenging boolean functions. Evol. Intell. **5**, 87–102 (2012)

22. Friedman, N.: Inferring cellular networks using probabilistic graphical models. Science. **303**, 799-805 (2004)
23. Gibson, D.G., Young, L., Chuang, R.Y., Venter, J.C., Hutchison, C.A., et al.: Enzymatic assembly of DNA molecules up to several hundred kilobases. Nat. Methods. **6**, 343-45 (2009)
24. Glaab, E., Bacardit, J., Garibaldi, J.M., Krasnogor, N.: Using rule-based machine learning for candidate disease gene prioritization and sample classification of cancer gene expression data. PLoS ONE. (2012) doi: 10.1371/journal.pone.0039932
25. Golberg, D.E.: Genetic algorithms in search, optimization, and machine learning. Addison-Wesley, Boston (1989)
26. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science. **313**, 504–507 (2006)
27. Hiroe, A., Tsuge, K., Nomura, C.T., Itaya, M., Tsuge, T.: Rearrangement of gene order in the *phaCAB* operon leads to effective production of ultrahigh-molecular-weight poly[(R)-3-Hydroxybutyrate] in genetically engineered *Escherichia coli*. Appl. Environ. Microbiol. (2012) doi: 10:1128/AEM.07715-11
28. Holland, J.H.: Adaptation in natural and artificial system: An introduction with application to biology, control and artificial intelligence, University of Michigan Press, Ann Arbor (1975)
29. Holland, J.H.: Adaptive algorithms for discovering and using general patterns in growing knowledge bases. Int. J. Pol. Anal. Inform. Syst. **4**, 245–268 (1980)
30. Holland, J.H.: Escaping brittleness: The possibilities of general-purpose learning algorithms applied to parallel rule-based systems. In: Michalski, R.S., Carbonell, J.G., Mitchell T.M. (eds.) Machine learning: An artificial intelligence approach, pp. 593623. Morgan Kaufmann, Los Altos (1986)
31. Holland, J.H., Reitman, J.S.: Cognitive systems based on adaptive algorithms. In: Waterman, D.A., Hayes-Roth, F. (eds.) Pattern directed inference systems, pp. 313-329. Academic Press, New York (1978)
32. Holmes, J.H., Lanzi, P.L., Stolzmann, W., Wilson, S.W.: Learning classifier systems: New models, successful applications. Inform. Process. Lett. (2002) doi: http://dx.doi.org/10.1016/S0020-0190(01)00283-6
33. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Netw. **2**, 359–366 (1989)
34. Isaacs, F.J., Carr, P.A., Wang, H.H., Lajoie, M.J., Sterling, B., et al. Precise manipulation of chromosomes *in vivo* enables genome-wide codon replacement. Science. **333**, 348-53 (2011)
35. Karlic, R., Chung, H.R., Lasserre, J., Vlahovicek, K., Vingron, M.: Histone modification levels are predictive for gene expression. P. Natl. Acad. Sci. USA. (2010) doi: 10.1073/pnas.0909344107
36. Kharbat, F., Odeh, M., Bull, L.: Knowledge discovery from medical data: An empirical study with XCS. In: Bull, L., Bernado-Mansilla, E., Holmes, J. (eds.) Learning classifier systems in data mining, pp. 93121. Springer, Heidelberg (2008)
37. Kovacs, T.: Strength or accuracy: credit assignment in learning classifier systems. Springer, London (2004)
38. Kushwaha, M., Salis, H.: A portable expression resource for engineering cross-species genetic circuits and pathways. Nat. Commun. (2015) doi: 10.1038/ncomms8832
39. Larman, H.B., Zhao, Z., Laserson, U., Li, M.Z., Ciccia, A., et al.: Autoantigen discovery with a synthetic human peptidome. Nat. Biotechnol. **29**, 535-541 (2011)
40. Lee, S.K., Chou, H., Ham, T.S., Lee, T.S., Keasling, J.D.: Metabolic engineering of microorganisms for biofuels production: from bugs to synthetic biology to fuels. Curr. Opin. Biotechnol. **19**, 556–563 (2008)
41. Li, M.Z., Elledge, S.J.: Harnessing homologous recombination *in vitro* to generate recombinant DNA via SLIC. Nat. Methods. **4**, 251-256 (2007)
42. Libbrecht, M.W., Noble, W.S.: Machine learning applications in genetics and genomics. Nat. Rev. Genet. (2015) doi: 10.1038/nrg3920
43. Lim, H.N., Lee, Y., Hussein, R.: Fundamental relationship between operon organization and gene expression. Proc. Natl. Acad. Sci. USA. (2011) doi: 10.1073/pnas.1105692108

44. Lu, T.K., Collins, J.J.: Dispersing biofilms with engineered enzymatic bacteriophage.Proc. Natl. Acad. Sci. USA. (2007) doi: 10.1073/pnas.0704624104
45. Lu, T.K., Collins, J.J.: Engineered bacteriophage targeting gene networks as adjuvants for antibiotic therapy. Proc. Natl. Acad. Sci. USA. (2009) doi: 10.1073/pnas.0800442106
46. Nakagawa, Y., Yugi, K., Tsuge, K., Itaya, M., Yanagawa, H., et al.: Operon structure optimization by random self-assembly. Nat. Comput. (2010) doi: 10.1007/s11047-009-9141-0
47. Nakata, M., Kovacs, T., Takadama, K.: A modified XCS classifier system for sequence labeling. In: Proceedings of the 2014 conference on Genetic and evolutionary computation, pp. 565-572. ACM Press, New York (2014)
48. Nishizaki, T., Tsuge, K., Itaya, M., Doi, N., Yanagawa, H.: Metabolic engineering of carotenoid biosynthesis in *Escherichia coli* by ordered gene assembly in *Bacillus subtilis*. Appl. Environ. Microbiol. (2007) doi: 10.1128/AEM.02268-06
49. Ouyang, Z., Zhou, Q., Wong, W.H.: ChIP-Seq of transcription factors predicts absolute and differential gene expression in embryonic stem cells. Proc. Natl. Acad. Sci. USA. (2009) doi: 10.1073/pnas.0904863106
50. Pfleger, B.F., Pitera, D.J., Smolke, C.D., Keasling, J.D.: Combinatorial engineering of intergenic regions in operons tunes expression of multiple genes. Nat. Biotechnol. (2006) doi: 10.1038/nbt1226
51. Quinlan, J.R.: C4.5: programs for machine learning. Morgan Kaufmann, San Mateo (1993)
52. Rabinovitch-Deere, C.A., Oliver, J.W., Rodriguez, G.M., Atsumi, S.: Synthetic biology and metabolic engineering approaches to produce biofuels. Chem. Rev. (2013) doi: 10.1021/cr300361t
53. Rochat, T., Bouloc, P., Repoila, F.: Gene expression control by selective RNA processing and stabilization in bacteria. FEMS Microbiol. Lett. (2013) doi: 10.1111/1574-6968.12162
54. Smith, S.F.: A learning system based on genetic adaptive algorithms. PhD thesis, University of Pittsburgh (1980)
55. Smith, S.F.: Flexible learning of problem solving heuristics through adaptive search. In: Proceedings of the Eighth International Joint Conference on Artificial Intelligence, pp. 421-425. Morgan Kaufmann, San Francisco (1983)
56. Sutton, R.S., Barto, A.G.: Reinforcement learning. MIT Press, Cambridge (1998)
57. Tsuge, K., Matsui, K., Itaya, M.: One step assembly of multiple DNA fragments with a designed order and orientation in *Bacillus subtilis* plasmid. Nucleic Acids Res. (2003) doi: 10.1093/nar/gng133
58. Urbanowicz, R.J., Moore, J.H.: Learning classifier systems: A complete introduction, review, and roadmap. Journal of Artificial Evolution and Applications. (2009) doi: 10.1155/2009/736398
59. Watkins, C.: Learning from delayed rewards. PhD thesis, University of Cambridge (1989)
60. Weber, W., Fussenegger, M.: Emerging biomedical applications of synthetic biology. Nat. Rev. Genet. (2012) doi: 10.1038/nrg3094
61. White, M.M.: Pretty subunits all in a row: using concatenated subunit constructs to force the expression of receptors with defined subunit stoichiometry and spatial arrangement. Mol. Pharmacol. **69**, 407-410 (2006)
62. Wilson, S.W.: ZCS: A zeroth level classifier system. Evol. Comput. **2**, 1-18 (1994)
63. Wilson, S.W.: Classifier fitness based on accuracy. Evol. Comput. **3**, 149-175 (1995)
64. Wright, C.M., Wright, R.C., Eshleman, J.R., Ostermeier, M.: A protein therapeutic modality founded on molecular regulation. Proc. Natl. Acad. Sci. USA. (2011) doi: 10.1073/pnas.1102803108
65. Xie, Z., Wroblewska, L., Prochazka, L., Weiss, R., Benenson, Y.: Multi-input RNAi-based logic circuit for identification of specific cancer cells. Science. **333**, 1307-1311 (2011)
66. Xu, C., Huang, R., Teng, L., Jing, X., Hu, J., et al.: Cellulosome stoichiometry in Clostridium cellulolyticum is regulated by selective RNA processing and stabilization. Nat. Comm. (2015) doi: 10.1038/ncomms7900
67. Yanisch-Perron, C., Vieira, J., Messing, J.: Improved M13 phage cloning vectors and host strains: Nucleotide sequences of the M13mp18 and pUC19 vectors. Gene. **33**, 103-19 (1985)
68. Zibakhsh, A., Abadeh, M.S.: Gene selection for cancer tumor detection using a novel memetic algorithm with a multi-view fitness function. Eng. Appl. Artif. Intell. **26**, 1274-1281 (2013)