Efficient Analysis of Homeostasis of Gene Networks with Compositional Approach

Sohei Ito[1,2], Kenji Osari[3], Shigeki Hagihara[4] and Naoki Yonezaki[5,6]

[1]*Department of Fisheries Distribution and Management, National Fisheries University, 2-7-1 Nagata-Honmachi, Shimonoseki, Yamaguchi, Japan*

[2]*School of Business Administration in Karviná, Silesian University in Opava, Univerzitní nám. 76, 733 40 Karviná, Czech Republic*

[3]*Yahoo Japan Corporation, 9-7-1 Akasaka, Minato-ku, Tokyo 107-6211, Japan*

[4]*Department of Computer Science, Tokyo Institute of Technology, 2-12-1 Ookayama, Meguro-ku, Tokyo 152-8550, Japan*

[5]*Graduate School of Information Environment, Tokyo Denki University, 2-1200, Muzai Gakuendai, Inzai-shi, Chiba 270-1382, Japan*

[6]*The Open University of Japan, 2-11 Wakaba, Mihama-ku, Chiba City, Chiba 261-8586, Japan*

Keywords: Gene Regulatory Network, Systems Biology, Homeostasis, Temporal Logic, Realisability.

Abstract: Homeostasis is an important property of life. Thanks to this property, living organisms keep their cellular conditions within an acceptable range to function normally. To understand mechanisms of homeostasis and analyse it, the systems biology approach is indispensable. For this purpose, we proposed a qualitative approach to model gene regulatory networks with logical formulae and formulate the homeostasis in terms of a kind of logical property – called *realisability* of linear temporal logic. This concise formulation of homeostasis naturally yields the method for analysing homeostasis of gene networks using realisability checkers. However, the realisability problem is well-known for its high computational complexity – double-exponential in the size of a formula – and the applicability of this approach will be limited to small gene networks, since the size of formula increases as the network does. To overcome this limitation, we leverage a compositional method to check realisability in which a formula is divided into a few sub-formulae. The difficulty in compositional approach is that we do not know how we obtain a good division. To tackle this issue, we introduce a new clustering algorithm based on a characteristic function on formulae, which calculates the size of formulae and the variation of propositions. The experimental results show that our method gives a good division to benefit from the compositional method.

1 INTRODUCTION

Since the end of the 20th century, the advances of experimental and high-throughput technologies in molecular biology have enabled us to produce huge amount of biological data. Compared to such advancement, the understanding of biological systems and how they work seems less understood than expected. As genes and proteins are components of the system, knowledge about each component does not give us how the system is working as entirety. Therefore, a new research field - *systems biology* - has emerged.

Systems biology aims to understand organisms as systems. Systems biology covers many topics such as structural identification of biological systems, construction of mathematical models which fit to observed phenomena, and development of modelling and analysing tools (Funahashi et al., 2003; Naldi et al., 2009; Helikar et al., 2012).

Usual approach in systems biology is to model a system with ordinary differential equations. Since there is uncertainty in kinetic parameters, we fit them to observed data to obtain a plausible model. However, the more components we have in the system, the more equations we have thus parameter fitting or computing analytical solution of the equations become almost infeasible. To overcome this difficulty, computational models have been used in systems biology (Fisher and Henzinger, 2007), which abstract real numerical behaviours into some discrete state sequences. There are several approaches to such computational models depending on underlying forma-

Ito S., Osari K., Hagihara S. and Yonezaki N.
Efficient Analysis of Homeostasis of Gene Networks with Compositional Approach.
DOI: 10.5220/0006093600170028
In *Proceedings of the 10th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2017)*, pages 17-28
ISBN: 978-989-758-214-1

lisms; Boolean networks (Thomas, 1991), Petri nets (Heiner et al., 2008), timed-automata (Batt et al., 2007) and process algebras (Ciocchetta and Hillston, 2009). The limitation of such computational models is that they require concrete molecular mechanisms, regulatory logics and sometimes a kinetic parameters. Since biological systems are incomplete in most cases, the applicability of such computational models are limited. In this regard, the constraint-based modelling is promising in which systems are described as a set of constraints (Palsson, 2000). We can model biological systems with incomplete information. The more information we have on the target system, the more constraints we have in a model.

The constraint-based modelling paradigm has a good connection with logical specification of *reactive systems* in which the system behaviour is described as a set of logical constraints (Mori and Yonezaki, 1993; Vanitha et al., 2000; Osari et al., 2014). Based on this close connection, we proposed a qualitative method for modelling and analysing gene networks (Ito et al., 2010; Ito et al., 2015b) using linear temporal logic (LTL). In this approach, we specify possible behaviours of networks by LTL as a set of logical constraints and analyse properties of networks (also written in LTL) by checking satisfiability of the formulae. This idea is based on the verification of reactive system specifications. Using this correspondence, we later formulated the notion of homeostasis by *realisability* of reactive system specifications (Pnueli and Rosner, 1989; Abadi et al., 1989) and demonstrated how homeostasis in gene networks can be analysed (Ito et al., 2014a).

The problem in our approach was the high-complexity of checking LTL realisability which is doubly-exponential in the size of a formula. As we reported in (Ito et al., 2015a), checking homeostasis of gene networks with even a few nodes takes several minutes. Since the sizes of formulae characterising possible behaviours of a network is proportional to the size of the network, we need to devise a method to mitigate the computational difficulty in the analysis of homeostasis.

To facilitate realisability checking, a *compositional* approach has been a promising method to analyse large formulae (Filiot et al., 2011) in which an LTL formula is divided into several sub-formulae that are analysed separately. The results are then merged to obtain the final outcome of the verification. The non-trivial problem in compositional analysis is how to divide a formula, which is critical to the performance of analysis.

The aim of this paper is to provide a method for finding good divisions as a front-end of a compositional analysis of homeostasis. Our approach to this problem is to develop a new clustering algorithm which clusters clauses (a piece of formula) based on a 'score' of clusters. The 'score' of clusters indicates how the clustering is good for compositional analysis. We implement our algorithm and show experimental results of analysing homeostasis of several gene networks. For larger gene networks, the gain from compositional approach is quite much: with compositional approach with our clustering front-end, we can verify in a few minutes, or even in a few seconds, networks that cannot be verified with monolithic approach in 1 hour.

The rest of the paper is organised as follows. Section 2 introduces LTL and the notion of *realisability* of reactive system specifications. Section 3 reviews how we model gene networks in LTL and formulates homeostasis by realisability. Section 4 introduces the compositional analysis of realisability. We show how we divide a network specifications to compositionally analyse homeostasis. Section 5 shows and discusses the experimental results of compositional analysis. In section 6, we discuss the relationship between our method and other similar approaches. Section 7 offers conclusion and discusses some future directions.

2 PRELIMINARY

2.1 Linear Temporal Logic

If A is a finite set, A^ω denotes the set of all infinite sequences on A. The i-th element of $\sigma \in A^\omega$ is denoted by $\sigma[i]$. Let AP be a set of propositions. A *time structure* is a sequence $\sigma \in (2^{AP})^\omega$ where 2^{AP} is the powerset of AP. The formulae of LTL are defined as follows.

- $p \in AP$ is a formula.
- If ϕ and ψ are formulae, then $\neg\phi, \phi \wedge \psi, \phi \vee \psi$ and $\phi U \psi$ are also formulae.

We introduce the usual abbreviations: $\bot \equiv p \wedge \neg p$ for some $p \in AP$, $\top \equiv \neg\bot$, $\phi \rightarrow \psi \equiv \neg\phi \vee \psi$, $\phi \leftrightarrow \psi \equiv (\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi)$, $F\phi \equiv \top U \phi$, $G\phi \equiv \neg F \neg \phi$, and $\phi W \psi \equiv (\phi U \psi) \vee G\phi$. We assume that $\wedge, \vee$ and U binds more strongly than $\rightarrow$ and unary connectives binds more strongly than binary ones.

Let σ be a time structure and ϕ be a formula. The semantics of LTL is defined by the relation $\sigma \models \phi$ representing ϕ is true in σ. The satisfaction relation $\models$ is defined as follows.

$$
\begin{array}{lll}
\sigma \models p & \text{iff} & p \in \sigma[0] \text{ for } p \in AP \\
\sigma \models \neg\phi & \text{iff} & \sigma \not\models \phi \\
\sigma \models \phi \wedge \psi & \text{iff} & \sigma \models \phi \text{ and } \sigma \models \psi \\
\sigma \models \phi \vee \psi & \text{iff} & \sigma \models \phi \text{ or } \sigma \models \psi \\
\sigma \models \phi U \psi & \text{iff} & (\exists i \geq 0)(\sigma^i \models \psi \text{ and } \\
 & & \forall j(0 \leq j < i)\sigma^j \models \phi)
\end{array}
$$

where $\sigma^i = \sigma[i]\sigma[i+1]\ldots$, i.e. the i-th suffix of σ. An LTL formula ϕ is *satisfiable* if there exists a time structure σ such that $\sigma \models \phi$. We also say that σ is a *model* of ϕ.

2.2 Reactive Systems and Realisability

A *reactive system* is defined as a triple $\langle X, Y, r\rangle$, where X is a set of events caused by the environment, Y is a set of events caused by the system and $r : (2^X)^+ \to 2^Y$ is a reaction function. The expression $(2^X)^+$ denotes the set of all finite sequences on subsets of X. A reaction function determines how the system reacts to environmental (or external) input sequences. Reactive system is a natural formalisation of systems which appropriately respond to requests from the environment. Systems controlling vending machines, elevators, air traffic and nuclear power plants are examples of reactive systems. Gene networks which respond to inputs or stimulation from the environment such as glucose increase, change of temperature or blood pressure can also be considered as reactive systems.

A specification of a reactive system stipulates how it responds to inputs from the environment. For example, for a controller of an elevator system, a specification will be e.g. 'if the open button is pushed, the door opens' or 'if a call button of a certain floor is pushed, the lift will come to the floor'.

Realisability is an important property of reactive system specifications (Pnueli and Rosner, 1989; Abadi et al., 1989). A realisable specification guarantees that there exists a reactive system which responds to any environmental input of any timing without violating the specification.

To check realisability of a reactive system specification, it should be described in a language with formal and rigorous semantics. LTL is known to be one of many other formal languages suitable for this purpose and several realisability checkers of LTL are available (Jobstmann et al., 2007; Filiot et al., 2009; Bloem et al., 2010).

Now we define the notion of realisability of LTL specifications. Let AP be a set of atomic propositions which is partitioned into X and Y. X corresponds to external events and Y to internal events. We denote a time structure σ on AP as $\langle x_0, y_0\rangle\langle x_1, y_1\rangle\ldots$ where $x_i \subseteq X$, $y_i \subseteq Y$ and $\sigma[i] = x_i \cup y_i$. Let ϕ be an LTL specification. We say $\langle X, Y, \phi\rangle$ is *realisable* if there exists a reactive system $RS = \langle X, Y, r\rangle$ such that

$$\forall \tilde{x}.behave_{RS}(\tilde{x}) \models \phi,$$

where $\tilde{x} \in (2^X)^\omega$ and $behave_{RS}(\tilde{x})$ is the infinite behaviour determined by RS, that is,

$$behave_{RS}(\tilde{x}) = \langle x_0, y_0\rangle\langle x_1, y_1\rangle\ldots,$$

where $\tilde{x} = x_0 x_1 \ldots$ and $y_i = r(x_0 \ldots x_i)$.

Intuitively a specification ϕ is realisable if there exists a system which controls its internal events in reaction to any sequence of external events, so that its behaviour satisfies the specification ϕ.

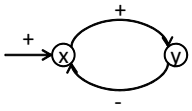

Figure 1: A gene network in which x and y are genes. Plus-edges represent activation relationship and minus-edges represent inhibition relationship. Gene x receives positive input from environment.

3 QUALITATIVE ANALYSIS OF HOMEOSTASIS IN GENE NETWORKS

In this section we review how we model the gene networks and analyse homeostasis of them by LTL.

3.1 Modelling Possible Behaviours of Gene Networks in LTL

Qualitative principles for characterising behaviours of a gene network are as follows:

- A gene is ON when its activators are expressed beyond some thresholds.
- A gene is OFF when its inhibitors are expressed beyond some thresholds.
- If a gene is ON, its expression level increases.
- If a gene is OFF, its expression level decreases.

To formally describe these principles in LTL, we introduce propositions for a given network which represent the state (or *configuration*) of a network, such as whether genes are ON and whether genes are expressed beyond certain thresholds.

We show how we describe behaviours of networks using an example network depicted in Fig. 1, where x and y represent genes, plus-edges mean activation and the minus-edge inhibition. Since gene x activates gene y, there exists a threshold expression level of x

above which gene x activates gene y. We write x_y for this threshold level. Similarly, gene y has the threshold y_x to inhibit x. Moreover, gene x receives the positive input from environment. We write e_x for the threshold of the input to activate x.

For this network we introduce the following propositions.

- on_x, on_y: whether gene x and y are ON respectively.
- x_y, y_x: whether gene x and y are expressed beyond the threshold x_y and y_x respectively[1].
- in_x: whether the input to x is ON.
- e_x: whether the positive input from the environment to x is beyond the threshold e_x.

Using these propositions, we specify the above qualitative principles in LTL. For example, the fact 'gene y is positively regulated by gene x' is described as

$$G(x_y \leftrightarrow on_y)$$

in LTL. Intuitively this formula says gene y is ON if, and only if, gene x is expressed beyond the threshold x_y (i.e. proposition x_y is true) due to positive effect of gene x toward gene y. As for gene x, it is negatively regulated by gene y and has positive input from the environment. A condition for activation and inactivation of such multi-regulated genes depends on a function which merges the multiple effects. We assume that gene x is ON if gene y is not expressed beyond the threshold y_x and, in addition, the input from the environment to gene x is beyond the threshold e_x; that is, the negative effect of gene y is not operating and the positive effect of the input is operating. Then this can be described as

$$G(e_x \wedge \neg y_x \rightarrow on_x).$$

This formula says that if the input level is beyond the threshold e_x and gene y is not expressed beyond the threshold y_x (i.e. proposition y_x is false; $\neg y_x$ is true), then gene x is ON.

If gene x is ON, the expression level of gene x is growing. In other words, it will reach the threshold x_y unless gene x becomes OFF prematurely. This fact is described as

$$G(on_x \rightarrow F(x_y \vee \neg on_x)).$$

If gene x is ON and expressed beyond the threshold x_y, it keeps the level until gene x is OFF since this is the largest threshold of gene x. This can be described as

$$G(on_x \wedge x_y \rightarrow x_y W \neg on_x).$$

[1]Note that the threshold levels are denoted in roman whereas the propositions corresponding to them are denoted in italic.

This formula means 'if gene x is ON and the current expression level of gene x is above x_y, gene x keeps its level as long as gene x is ON'.

If gene x is OFF, its transcription product decreases due to degradation. We have the symmetric formulae to the case of gene x being ON:

$$G(\neg on_x \rightarrow F(\neg x_y \vee on_x)),$$
$$G(\neg on_x \wedge \neg x_y \rightarrow \neg x_y W on_x).$$

We have similar clauses for expression of gene y.

We do not thoroughly explain how we describe constraints which model the possible behaviours of gene networks. Interested readers may wish to consult (Ito et al., 2015b) for detail.

3.2 Analysing Homeostasis by Realisability Checking

Biological homeostasis is informally defined as the tendency of a system to maintain its internal stability or function against *any* situation or stimulus. This property is closely related to realisability since it says that there is a system which responds to *any* environmental inputs of any timing while keeping its specified internal conditions. Using this correspondence we formulate *homeostasis* by *realisability*.

In the previous section, we showed that a gene network can be modelled by an LTL formula which is the conjunction of clauses. Precisely speaking, the specification of a given network is a triple $\langle E, I, \varphi \rangle$ where

- E: the set of external propositions (inputs),
- I: the set of internal propositions (genes and thresholds),
- φ: the formula which characterises the possible behaviours of a given network.

We also write a biological property or function of the network, which we are to check whether the given network maintains it in response to any external input sequence. The examples of such property will be 'the expression level of a certain gene is within a tolerable range', 'once a gene becomes ON, its expression will be suppressed afterwards', et cetera. Such properties can be easily described in LTL.

In realistic situation, we sometimes put an assumption (or constraints) on the environment which restricts the input sequence. For example, we may consider the case that an input is oscillating, an input is always coming, or a certain input comes only after another comes, and so on. Sometimes it is useful to consider homeostasis of a system under such environmental assumptions. We also describe such assumptions in LTL.

Now we define homeostasis of a gene network with respect to a property under an environmental assumption. The following definition is an extended version of our previous definition of homeostasis (Ito et al., 2014a) in that we include environmental assumption ξ.

Definition 1. *A property ψ is* homeostatic *with respect to a behaviour specification $\langle E, I, \varphi \rangle$ under the environmental assumption ξ if $\langle E, I, \xi \rightarrow \varphi \wedge \psi \rangle$ is realisable.*

Note that a homeostasis of a gene network is defined with a certain biological property. A network which is homeostatic with respect to a certain property has a strategy to respond for any input sequence which satisfies ξ, without violating neither behaviour constraint φ nor the property ψ. We can set ξ as true (i.e. empty clause), which amounts to put no assumption on environment.

Example 1. *The network depicted in Fig. 1 is homeostatic with respect to a property 'whenever gene x becomes ON, the expression of gene x will be suppressed afterwards'. This property is described in LTL as follows:*

$$G(on_x \rightarrow F(\neg on_x \wedge \neg x_y)).$$

Let φ be a behavioural specification of the network (partly shown in the previous section). This is verified by checking realisability of the specification $\langle \{in_x\}, \{on_x, on_y, x_y, y_x, e_x\}, \varphi \wedge G(on_x \rightarrow F(\neg on_x \wedge \neg x_y)) \rangle$.

This property is also homeostatic if we put an environmental assumption 'the input to x is always ON' which is described as:

$$Gin_x.$$

That is to say, the specification $\langle \{in_x\}, \{on_x, on_y, x_y, y_x, e_x\}, Gin_x \rightarrow \varphi \wedge \psi \rangle$ is realisable (where ψ represents the above property).

Since we defined homeostasis by realisability, we can use realisability checkers to conduct such analysis. An obstacle with this framework is the high-complexity of LTL realisability problem which is 2EXPTIME-complete in the size of a formula (Pnueli and Rosner, 1989). Since the size of a network specification is proportional to the size of a network, the analysis of a larger network will be intractable in general. The next section discusses the compositional analysis method to mitigate this difficulty.

4 COMPOSITIONAL ANALYSIS OF HOMEOSTASIS

Recent advances in realisability checking technique enable us to handle large specifications. Since our framework to analyse homeostasis uses realisability checking, we can reap the benefit from the advances. Here we leverage the *compositional* algorithm in realisability checking (Filiot et al., 2011). In our setting, the compositional analysis of homeostasis of gene networks is stated as follows:

1. For a given specification $\langle E, I, \xi \rightarrow \bigwedge_{1 \leq i \leq n} \varphi_i \rangle$, compute a 'good' clustering of the formula $\{c_1, \ldots, c_k\}$, where each c_ℓ consists of φ_is, e.g. network specifications and biological properties.
2. We check the realisability of each sub-specification $\langle E, I, \xi \rightarrow \bigwedge_{\varphi \in c_\ell} \varphi \rangle (\ell \in \{1, \ldots, k\})$.
3. We merge the result of individual results.

Step 2 and 3 can be automatically done by the tool Acacia+ (Bohy et al., 2012). The problem is how we obtain a 'good' clustering of a given specification. Before proceeding, we first review the algorithm of Acacia+ to solve realisability compositionally.

4.1 Compositional Approach to Realisability Problems

The algorithm implemented in Acacia+ to solve realisability of LTL formula φ is as follows:

1. Translate φ to a universal co-Büchi automaton A_φ.
2. Translate A_φ to a safety game $G(A_\varphi)$.
3. Compute the winning strategy of the game $G(A_\varphi)$.

It is reported that the first step, i.e. translating an LTL formula to an equivalent automaton, is the bottleneck of the algorithm. Known algorithms to translate LTL formulae into equivalent automata run in exponential time with respect to the sizes of formulae. Especially for large LTL formulae the first step does not finish. To overcome this problem, the compositional approach is proposed. Large LTL formulae are often written as $\varphi = \varphi_1 \wedge \varphi_2 \wedge \cdots \wedge \varphi_n$. Thus in compositional approach, each sub-formula φ_i is translated into automaton A_{φ_i} and thus translated into a local game $G(A_{\varphi_i})$, then the winning strategy is computed for each $G(A_{\varphi_i})$. Thanks to the nice property of safety games, the winning strategy for $G(A_\varphi)$, the strategy for the original game, can be computed from the winning strategies for each local game $G(A_{\varphi_i})$.

In general, the specification is of the form $\xi \rightarrow \varphi$ where ξ is an environmental assumption. Therefore ξ

is distributed to each φ_i, that is, we solve the realisability of each $\xi \to \varphi_i$ separately.

The merit of this compositional approach is that we can reduce the time of translating formulae into equivalent automata, since each φ_i is much smaller than the original formula φ. Note that, however, we have extra cost of merging the winning strategies for local games, compared to non-compositional (i.e. monolithic) approach. For compositional approach to be useful, the reduction in automata translation must prevail the additional cost of merging the local winning strategies. Thus the performance of compositional approach depends on how we divide φ into several φ_is.

4.2 Applying to Our Problem

Since behaviour specifications for gene networks are written as conjunctions of clauses (see section 3.1), we can apply the compositional approach introduced in the previous section. Our specification in general consists of many clauses, thus it is not efficient to solve every single clauses separately as the overhead of integrating the winning strategies of the local games increases according to the number of local games.

Thus the possible approach is to distribute the clauses into several clusters $c_1, c_2, \ldots, c_k$, i.e. $\varphi = (\bigwedge c_1) \wedge (\bigwedge c_2) \wedge \cdots \wedge (\bigwedge c_k)$ where $\bigwedge c_i = \bigwedge_{\phi \in c_i} \phi$. The problem is how to distribute them. What is a good clustering?

For example, suppose that we have 20 clauses and distributing them into 2 clusters. It seems better to put 10 clauses into each cluster than to put 1 clause into one cluster and 19 clauses into the other. In such uneven clustering, the 19 clauses cluster will be a bottleneck and the efficiency may not improve as a whole. Thus a good clustering distributes clauses into clusters as equally as possible. However, the number of clauses may not be a good criterion, since the sizes of formulae are not the same; in an extreme situation, the size of a certain formula might be equal to the size of the rest. Thus we distribute clauses into clusters whose sizes are as equal as possible.

To facilitate winning strategy computation of each local game, it is desirable to obtain small automata. The size of automata A_φ is determined by the number of sub-formulae in φ. To compute the number of sub-formulae for each clause is not realistic because it is exponential to the size of the formula. The plausible criterion is the number of variation of propositions – the more variation of propositions we have, the more the number of sub-formulae. Thus we also take the variation of propositions in a cluster into consideration, in addition to the size of a cluster.

Now we formalise the above idea. Let $C = \{c_1, \ldots, c_k\}$ be a clustering of a specification $\langle E, I, \xi \to \bigwedge_{1 \le i \le n} \varphi_i \rangle$, that is, each cluster c_ℓ consists of φ_is. Note that the environmental constraint ξ is copied to each cluster. Let $N(\chi)$ and $V(\chi)$ respectively denote the number and the variations of propositions in a formula χ. We introduce the evaluation function $\mathcal{F}$ of clustering C as follows:

$$\mathcal{F}(C) = \sum_{1 \le i \le k} \mathcal{E}(\xi \to \bigwedge_{\varphi \in c_i} \varphi),$$
$$\text{where } \mathcal{E}(\chi) = N(\chi)^2 + V(\chi)^2$$

The function $\mathcal{E}$ represents the evaluation of a single formula. Due to the squared terms in the function $\mathcal{E}$, we can easily see that a clustering consists of two clusters of the sizes 5 and 5 is better than that of 9 and 1. A good clustering is the one which minimises the value of this function.

The number of possible clustering of n formulae into k clusters is a $S(n,k)$, i.e. the Stirling number of the second kind. Thus the naïve algorithm to find the optimal cluster is not realistic in general. Thus we introduce a suboptimal algorithm whose complexity is $O(kn^2)$ which we show in Algorithm 1.

Algorithm 1: Suboptimal clustering algorithm.

Input: $\{\varphi_1, \ldots, \varphi_n\}$ (specification), k (the number of the clusters)
Output: $C = \{c_1, \ldots, c_k\}$

```
c_1 ← {φ_1, ..., φ_n}
for i = 2 to k do
    c_i ← ∅
end for
C ← {c_1, ..., c_k}
min ← F(C)
repeat
    for all φ ∈ c_1 do
        change ← false
        for i = 2 to k do
            C' ← {c_1 \ {φ}, ..., c_i ∪ {φ}, ..., c_k}
            f ← F(C')
            if f < min then
                C ← C'
                min ← f
                change ← true
            end if
        end for
    end for
until change = false
```

This algorithm starts with the cluster c_1 with $\{\varphi_1, \ldots, \varphi_n\}$ and the others with empty sets. For each

step we move one φ in c_1 to another cluster c_i and evaluate the new clustering by function $\mathcal{F}$. If the evaluation decreases for some i, we adopt the minimal one among such clusterings. We repeat this process until moving φ from c_1 to any other clusters no longer decreases the evaluation. Since the outermost loop is executed at most n times, and each step decrements the number of elements of c_1, this algorithm runs in time $O(kn^2)$. We implemented the algorithm in OCaml and confirmed that this suboptimal algorithm computes a clustering whose evaluation is as good as the optimal clustering.

5 EXPERIMENTAL RESULTS AND DISCUSSION

This section discusses the experimental results of homeostasis analysis performed on several networks. Since compositional analysis is available for only realisable specifications, all specifications used in this experiment are realisable.

To solve the realisability of an LTL specification φ, we first construct an ω-automaton which is equivalent to φ, then construct a game on the automaton and compute a winning strategy of the system which can respond any environmental requests. Compositional analysis decreases the cost of constructing ω-automata from LTL formulae since the formula is divided into several small sub-formulae. However, we have extra cost of merging all winning strategies computed for each sub-formulae. Therefore, increasing the number of clusters may not necessarily reduce the cost of analysis as a whole. However, to predict the optimal number of clustering *a priori* is difficult. We, hence, demonstrate our clustering method with different number of clusters to see the impact of the number of clusters in our compositional method.

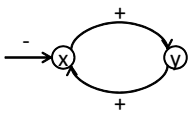

Figure 2: A bistable switch (Ito et al., 2014a).

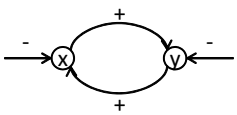

Figure 3: A bistable switch with additional inputs (Ito et al., 2014a).

The result of experiments are shown in table 1. These experiments are carried out on a computer with Intel Core i7-3820 3.60GHz CPU and 32GB memory. The realisability checker used is Acacia+ (Bohy et al., 2012). The network 'bistable switch' is the network that consists of two genes x and y where x activates y, y activates x and x receives negative input from environment (Fig. 2). The network 'bistable switch2' is the same as 'bistable switch' except both gene x and y receive negative inputs from environment (Fig. 3). The networks 'anti-stress response (a)(b)(c)' are the stress response networks studied in (Zhang and Andersen, 2007) (Fig. 4). The networks '3 genes' to '6 genes' are depicted in Figs. 5 to 8. The homeostatic properties we analysed are tendency to ease stress (described as $G(stress \rightarrow F\neg stress)$) for anti-stress response networks and stability of a certain gene expression (described as *Gon*) for others. As environmental assumptions, we put every input is oscillating; it is written as $G((input \rightarrow F\neg input) \wedge (\neg input \rightarrow Finput))$.

As we can see from the result, we benefit from compositional analyses compared to the monolithic approach ($k = 1$). The improvements in analyses of specifications from '3 genes' to '6 genes' are remarkable.

Concerning the number of clusters, as we stated at the beginning of this section, increasing the number of clusters does not necessarily reduce the entire verification time. We can see it from 'anti-stress response(c)' where the cases for $k > 4$ are worse than the monolithic approach. Even for larger specifications, the best number of clusters seems to be 2 or 3.

The computational time of our clustering algorithm is less than 0.01 regardless of the number of clusters for all networks except for '6genes' at $k = 6$ (0.11 seconds). This means that we can ignore the cost of computing clusters to use compositional analysis even for small networks such as 'bistable switch'. Note that table 1 includes the clustering times.

Now we compare our method to random clustering. For each specification, we generate 50 random clusterings. We first randomly choose the size (i.e. the number of clauses) of each cluster, then we randomly distribute clauses to each cluster according to the size.

We show the results of random clustering in table 2. When calculating average time of verification, the clusterings which cannot be verified within 1 hour are treated as 1 hour. The mark $\star$ in the table shows that there were such clusterings. Thus the true average is greater than the shown value.

For larger specifications such as 'bistable switch2', '3genes', '4genes' and '5genes', we can clearly see that our method outperforms random clustering. Readers might notice that for the network of

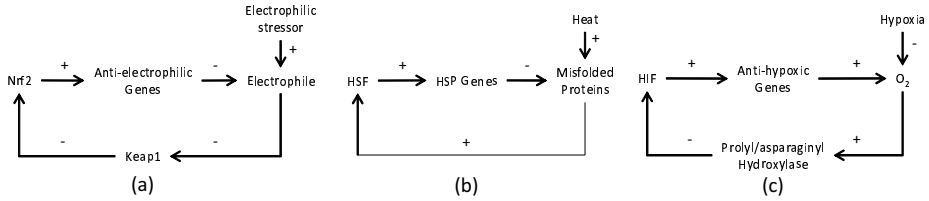

Figure 4: Anti-stress networks.

Table 1: Experimental results. Columns 'E' and 'I' respectively show the numbers of external propositions and internal propositions. Column 'S' shows the size of formula. (i.e. number of connectives and propositions). The columns 'Time(s)' show the total time of the verification for various number of clusters (k). '$k = 1$' means non-compositional analysis. For '$k > 2$ the times include computation of clustering.

				Time(s)					
Network	E	I	S	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$
bistable switch	1	6	232	0.40	0.07	0.08	0.10	0.10	0.07
bistable switch2	2	8	353	206.81	0.81	0.41	0.42	0.51	0.53
anti-stress response (a)	1	9	289	0.48	0.31	0.35	0.41	0.47	0.51
anti-stress response (b)	1	7	237	0.22	0.12	0.13	0.15	0.16	0.18
anti-stress response (c)	1	9	288	0.39	0.31	0.38	0.41	0.47	0.51
3 genes	3	10	418	1617.87	5.03	5.43	6.08	5.82	6.30
4 genes	2	12	444	> 3600	4.07	4.48	5.16	5.69	6.21
5 genes	3	15	547	> 3600	173.31	123.80	192.17	135.77	194.75
6 genes	3	18	646	> 3600	2676.37	2792.61	2877.03	2942.62	3087.91

Table 2: Results of random clustering. It shows average verification times of 50 randomly clustered specs. The figures show the time (in seconds) to check homeostasis. The stars on the figures show that some trials failed due to time out of 1 hour. Such clusterings are treated as finished in 3600 seconds for the sake of expedience.

Network	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$
bistable switch	0.11	0.11	0.11	0.11	0.12
bistable switch2	12.20	4.39	2.24	1.62	1.23
anti-stress response (a)	0.36	0.38	0.43	0.44	0.47
anti-stress response (b)	0.13	0.14	0.15	0.16	0.17
anti-stress response (c)	0.35	0.37	0.40	0.43	0.46
3 genes	81.80	29.44	50.88	10.44	10.94
4 genes	687.47*	441.53*	166.43	45.99	130.66
5 genes	1746.24*	845.6*	772.39*	841.6*	842.74*
6 genes	3220.6*	2953.01*	2767.81*	2830.68*	2641.21*

'6genes', the random clustering seems to outperform our method for some k. However, this is because we treated timed-out samples as 3600 seconds when averaging the results. Though we do not know the true average, it will be worse than the results of our method.

For smaller specifications, such as 'bistable switch' and 'anti-stress response' networks, our method is not clearly better than random clustering. The reason is that the formulae are not so large that the automata construction is not a heavy task in the verification process. For such specifications our strategy – to decrease the time of automata construction – may not be suitable.

Readers might wonder why '4genes' for $k = 5$ is considerably smaller than others in random clustering. Plausible explanation is that the random sampling happened to give biased clusterings. What is important here is that even for such biased random sampling, our method is much better.

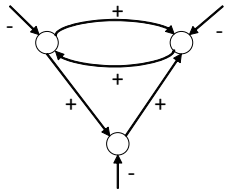

Figure 5: A network consisting of 3 genes.

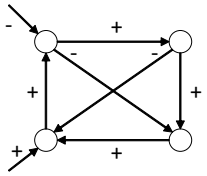

Figure 6: A network consisting of 4 genes.

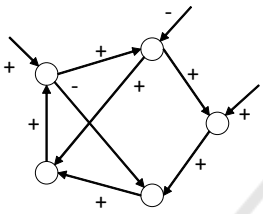

Figure 7: A network consisting of 5 genes.

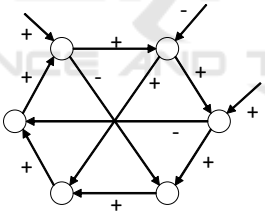

Figure 8: A network consisting of 6 genes.

We also show the standard deviations of the random clustering in table 3. As we can see from the table, the standard deviations are high for larger specifications. This means that our evaluation function is actually a statistically meaningful criterion, because random clustering ranges from better to worse. Our clustering algorithm tactfully chooses better ones.

We show another proof of the benefit of our algorithm: relative standings of our results (the values shown in Table 1) within the run-times of random clusterings. In other words, the percentile ranks of our results in the random clusterings. Table 4 shows them. As we can see, for $k = 2$ or 3 our algorithm gives good scores.

However, this table also shows that for $k \geq 4$ our evaluation function might not be a good criterion. This is because if we increase the number of clusters, the size of formulae in each cluster tends to be small, so that the automata construction time decreases and becomes less important in the verification process. Thus the larger k tends to impair the benefit of our method. Rather, we only reap the overhead of merging the winning strategies of the local games. This suggests another tactics for clustering: we focus on decreasing the computation time of winning strategy. We leave this issue for future work.

Before closing this section, we make a brief remark on a treatment of unrealisable specifications. As we mentioned at the beginning of this section, compositional approach is only applicable for realisable specification (Filiot et al., 2011; Bohy et al., 2012). In the current implementation of Acacia+, a user chooses options whether he checks realisability or unrealisability (or both in parallel). Thus when we are to verify a specification for which we do not know whether it is realisable or not, first we try realisability checking compositionally. If the verifier cannot prove realisability, we try unrealisability checking monolithically. However, since Acacia+ uses heuristic in which a certain bound is set when searching winning strategies, sometimes neither realisability nor unrealisability is proved. If this is the case, we set a larger bound and repeat the process until we have a definite result.

We summarise this section. For larger specifications, our method works well because the bottleneck in the verification process is the automata construction. If the number of clusters is small (2 or 3), our method is especially effective. If we increase the number of clusters, the bottleneck seems to shift from automata construction to winning strategy computation. To investigate a method to reduce the time for winning strategy computation is an interesting future work.

6 RELATED WORK

Sensitivity or robustness of biological systems is close but different from homeostasis. It analyses whether a property holds for a similar degree/probability when a parameter is modified, while homeostasis means that a given property holds for a certain set of input sequences. (Rizk et al., 2009) defines robustness of a biological system by measuring the deviation of the 'satisfaction degree' of an LTL formula (Fages and Rizk, 2008) caused by perturbations on parameters. They can treat the initial value of an environmental input as a parameter, thus homeostasis is partially covered. (Donzé et al., 2011) is a similar approach to (Rizk et al., 2009) for robustness analysis. They analyse continuous trajectories by means of signal tem-

Table 3: Standard deviation of the results of random clustering.

Network	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$
bistable switch	0.05	0.03	0.02	0.01	0.01
bistable switch2	23.00	12.1	4.51	3.74	2.30
anti-stress response (a)	0.09	0.08	0.13	0.04	0.04
anti-stress response (b)	0.02	0.02	0.01	0.01	0.01
anti-stress response (c)	0.06	0.09	0.03	0.04	0.05
3 genes	183.01	138.24	273.49	13.04	22.63
4 genes	1239.13*	1021.42*	618.16	218.94	377.97
5 genes	1593.59*	1318.82*	1183.29*	1295.33*	1270.92*
6 genes	700.55*	763.89*	835.77*	819.86*	989.15*

Table 4: Relative standing of the verification time with our method within the run-times of random clusterings. The value 'best' means that our method was the best.

Network	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$
bistable switch	10%	14%	39%	43%	best
bistable switch2	38%	13%	15%	41%	41%
anti-stress response (a)	27%	17%	64%	88%	88%
anti-stress response (b)	31%	45%	70%	70%	92%
anti-stress response (c)	23%	86%	68%	92%	92%
3 genes	14%	26%	47%	27%	27%
4 genes	17%	8%	34%	38%	43%
5 genes	19%	12%	43%	15%	35%
6 genes	14%	26%	41%	44%	50%

poral logic (STL) and its satisfaction-degree. In their approaches, it is unclear how to model arbitrary input scenario and assess the effect of it. SReach (Wang et al., 2015) is a tool to analyse stochastic hybrid systems, which can be used to model biological systems. SReach focuses the probabilistic reachability, that is, a property whether a given system reaches safe (or unsafe) states within a given probability range. SReach can conduct sensitivity analysis: Are the results of reachability analysis the same for different possible values of a certain system parameter? Since SReach only considers probabilities on state transitions, it is not clear how to model perturbations on inputs.

(Zhang and Andersen, 2007) analyses homeostasis of anti-stress gene regulatory networks by means of control theory. They consider steady-state response curves of anti-stress genes to a dose of stress. Their control-theoretical method is tailored to anti-stress networks and feasible to predict the effect of changing local response coefficients, but does not provide a generic framework applicable to any network.

In contrast to above approaches, our approach is purely qualitative. We do not need any kinetic parameters, which is usually difficult to obtain. There have been proposed several qualitative approaches to model and analyse biological systems based on several different formalisms - BIOCHAM (Fages et al., 2004) based on rewriting system, SMBioNet (Bernot et al., 2004) and Qualitative networks (Schaub et al., 2007) based on extended Boolean networks, and GNA (de Jong et al., 2003) based on piecewise linear differential equation. Compared to these formalisms, our LTL-based model of gene networks are flexible and easy to construct since the basic principle to model the behaviours are quite simple. The change of condition or assumption (e.g. bias of multivariate regulation) is immediate. Although the above tools are useful for checking whether a biological property can be true in network behaviours, it is unknown how to define and analyse homeostasis. As to the size of analysable networks, we cannot directly compare the scalability since the properties analysed and behaviours modelled in these tools are different and the cost of analysis also depends on the complexity of the network structure as well as the size. Furthermore, the examples demonstrated are very few and we do not have canonical benchmarks in this field. For information, we refer the size of networks analysed in these

tools; SMBioNet analyses a 3 nodes network, GNA a 10 nodes network and Qualitative networks a 2160 nodes network (in 10 hours). Qualitative network can handle very large networks compared to others including our framework, but we should keep in mind that Qualitative networks only consider propositional properties on steady states of the network behaviour, i.e. a temporal property such as 'when a gene is activated it will be suppressed later' cannot be checked. Furthermore, the behaviour of a Qualitative network is *deterministic*. In contrast we consider all *possible* network behaviours and their temporal properties with responses to environmental inputs.

As for compositional analysis of gene networks, we proposed to divide a network into several subnetworks and verify them individually (Ito et al., 2013; Ito et al., 2015b). This means we manually divide an LTL formula into several sub-formulae, considering the network structure. We have not discussed any algorithm to divide a network and the corresponding formula. In this paper, our method is based on a syntactic structure of a formula, not on a network structure. Thus it is easy to cluster a formula algorithmically.

7 CONCLUSION

In this paper we proposed compositional analysis of homeostasis of gene networks. We introduced a new clustering algorithm based on characteristic function on LTL formulae to divide a network specification. The experimental results show that our method drastically improves the performance in analysing larger network specifications. Our clustering algorithm is not limited to analyse homeostasis of gene networks but for realisability checking of any reactive system specification.

All examples in our experiments are carried out to see the impact of the sizes of networks. Thus the homeostatic properties are simple and described as relatively small formulae compared to network specifications. We would like to assess the impact of homeostatic properties to be more confident in our approach.

For further improvement, we are interested in importing Ito et al.'s approximate method (Ito et al., 2014b) to check homeostasis of gene networks. Another important future direction is to analyse real networks which may now be feasible thanks to this work, and elucidate regulation mechanisms contributing biological homeostasis.

REFERENCES

Abadi, M., Lamport, L., and Wolper, P. (1989). Realizable and unrealizable specifications of reactive systems. In *ICALP '89: Proceedings of the 16th International Colloquium on Automata, Languages and Programming*, volume 372 of *LNCS*, pages 1–17, London, UK. Springer-Verlag.

Batt, G., Salah, R. B., and Maler, O. (2007). On timed models of gene networks. In *FORMATS 2007*, volume 4763 of *LNCS*, pages 38–52.

Bernot, G., Comet, J., Richard, A., and Guespin, J. (2004). Application of formal methods to biological regulatory networks: extending Thomas' asynchronous logical approach with temporal logic. *J. Theor. Biol.*, 229(3):339–347.

Bloem, R., Cimatti, A., Greimel, K., Hofferek, G., Könighofer, R., Roveri, M., Schuppan, V., and Seeber, R. (2010). RATSY – a new requirements analysis tool with synthesis. In *Proceedings of the 22nd international conference on Computer Aided Verification*, volume 6174 of *LNCS*, pages 425–429, Berlin, Heidelberg. Springer-Verlag.

Bohy, A., Bruyère, V., Filiot, E., Jin, N., and Raskin, J.-F. (2012). Acacia+, a tool for LTL synthesis. In *Proceedings of the 24th International Conference on Computer Aided Verification*, CAV'12, pages 652–657.

Ciocchetta, F. and Hillston, J. (2009). Bio-PEPA: A framework for the modelling and analysis of biological systems. *Theor. Comput. Sci.*, 410:3065–3084.

de Jong, H., Geiselmann, J., Hernandez, G., and Page, M. (2003). Genetic network analyzer: Qualitative simulation of genetic regulatory networks. *Bioinformatics*, 19(3):336–344.

Donzé, A., Fanchon, E., Gattepaille, L. M., Maler, O., and Tracqui, P. (2011). Robustness analysis and behavior discrimination in enzymatic reaction networks. *PLOS One*, 6(9):e24246.

Fages, F. and Rizk, A. (2008). On temporal logic constraint solving for analyzing numerical data time series. *Theor. Comput. Sci.*, 408(1):55–65.

Fages, F., Soliman, S., and Chabrier-Rivier, N. (2004). Modelling and querying interaction networks in the biochemical abstract machine BIOCHAM. *J. Biol. Phys. Chem.*, 4:64–73.

Filiot, E., Jin, N., and Raskin, J.-F. (2009). An antichain algorithm for LTL realizability. In *Proceedings of the 21st International Conference on Computer Aided Verification*, volume 5126 of *LNCS*, pages 263–277, Berlin, Heidelberg. Springer-Verlag.

Filiot, E., Jin, N., and Raskin, J.-F. (2011). Antichains and compositional algorithms for LTL synthesis. *Formal Methods in System Design*, 39(3):261–296.

Fisher, J. and Henzinger, T. (2007). Executable cell biology. *Nat. Biotechnol.*, 25(11):1239–1249.

Funahashi, A., Tanimura, N., Morohashi, M., and Kitano, H. (2003). Celldesigner: a process diagram editor for gene-regulatory and biochemical networks. *BIOSILICO*, 1:159–162.

Heiner, M., Gilbert, D. R., and Donaldson, R. (2008). Petri nets for systems and synthetic biology. In *SFM 2008*, volume 5016 of *LNCS*, pages 215–264.

Helikar, T., Kowal, B., McClenathan, S., Bruckner, M., Rowley, T., Madrahimov, A., Wicks, B., Shrestha, M., Limbu, K., and Rogers, J. A. (2012). The cell collective: Toward an open and collaborative approach to systems biology. *BMC Systems Biology*, 6(1):96.

Ito, S., Hagihara, S., and Yonezaki, N. (2014a). A qualitative framework for analysing homeostasis in gene networks. In *Proceedings of BIOINFORMATICS 2014*, pages 5–16.

Ito, S., Hagihara, S., and Yonezaki, N. (2015a). Formulation of homeostasis by realisability on linear temporal logic. In Plantier, G., Schultz, T., Fred, A., and Gamboa, H., editors, *Biomedical Engineering Systems and Technologies: 7th International Joint Conference, BIOSTEC 2014, Angers, France, March 3-6, 2014, Revised Selected Papers*, pages 149–164. Springer International Publishing, Cham. http://dx.doi.org/10.1007/978-3-319-26129-4_10.

Ito, S., Ichinose, T., Shimakawa, M., Izumi, N., Hagihara, S., and Yonezaki, N. (2013). Modular analysis of gene networks by linear temporal logic. *J. Integrative Bioinformatics*, 10(2). http://dx.doi.org/10.2390/biecoll-jib-2013-216.

Ito, S., Ichinose, T., Shimakawa, M., Izumi, N., Hagihara, S., and Yonezaki, N. (2014b). Formal analysis of gene networks using network motifs. In Fernández-Chimeno, M., Fernandes, L. P., Alvarez, S., Stacey, D., Solé-Casals, J., Fred, A., and Gamboa, H., editors, *Biomedical Engineering Systems and Technologies: 6th International Joint Conference, BIOSTEC 2013, Barcelona, Spain, February 11-14, 2013, Revised Selected Papers*, pages 131–146. Springer Berlin Heidelberg, Berlin, Heidelberg. http://dx.doi.org/10.1007/978-3-662-44485-6_10.

Ito, S., Ichinose, T., Shimakawa, M., Izumi, N., Hagihara, S., and Yonezaki, N. (2015b). Qualitative analysis of gene regulatory networks by temporal logic. *Theor. Comput. Sci.*, 594(23):151–179. http://dx.doi.org/10.1016/j.tcs.2015.06.017.

Ito, S., Izumi, N., Hagihara, S., and Yonezaki, N. (2010). Qualitative analysis of gene regulatory networks by satisfiability checking of linear temporal logic. In *Proceedings of BIBE 2010*, pages 232–237. http://dx.doi.org/10.1109/BIBE.2010.45.

Jobstmann, B., Galler, S., Weiglhofer, M., and Bloem, R. (2007). Anzu: a tool for property synthesis. In *Proceedings of the 19th international conference on Computer aided verification*, volume 4590 of *LNCS*, pages 258–262, Berlin, Heidelberg. Springer-Verlag.

Mori, R. and Yonezaki, N. (1993). Several realizability concepts in reactive objects. In *Information Modeling and Knowledge Bases IV*, pages 407–424.

Naldi, A., Berenguier, D., Fauré, A., Lopez, F., Thieffry, D., and Chaouiya, C. (2009). Logical modelling of regulatory networks with ginsim 2.3. *Biosystems*, 97:134–139.

Osari, K., Murooka, T., Hagiwara, K., Ando, T., Shimakawa, M., Ito, S., Hagihara, S., and Yonezaki, N. (2014). An object-oriented language for parameterised reactive system specification based on linear temporal logic. In *Theory and Practice of Computation*, pages 121–143. WORLD SCIENTIFIC.

Palsson, B. (2000). The challenges of in silico biology. *Nat. Biotechnol.*, 18:1147–50.

Pnueli, A. and Rosner, R. (1989). On the synthesis of a reactive module. In *POPL '89: Proceedings of the 16th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 179–190, New York, NY, USA. ACM.

Rizk, A., Batt, G., Fages, F., and Soliman, S. (2009). A general computational method for robustness analysis with applications to synthetic gene networks. *Bioinformatics*, 25(12):i169–i178.

Schaub, M. A., Henzinger, T. A., and Fisher, J. (2007). Qualitative networks: A symbolic approach to analyze biological signaling networks. *BMC Systems Biology*, 1:4.

Thomas, R. (1991). Regulatory networks seen as asynchronous automata: A logical description. *J. Theor. Biol.*, 153(1):1–23.

Vanitha, V., Yamashita, K., Fukuzawa, K., and Yonezaki., N. (2000). A method for structuralisation of evolutional specifications of reactive systems. In *ICSE 2000, The Third International Workshop on Intelligent Software Engineering (WISE3)*, pages 30 – 38.

Wang, Q., Zuliani, P., Kong, S., Gao, S., and Clarke, E. M. (2015). SReach: A probabilistic bounded delta-reachability analyzer for stochastic hybrid systems. In Roux, O. and Bourdon, J., editors, *CMSB 2015*, volume 9308 of *LNCS*, pages 15–27, Cham. Springer International Publishing.

Zhang, Q. and Andersen, M. E. (2007). Dose response relationship in anti-stress gene regulatory networks. *PLoS Comput. Biol.*, 3(3).