A population-based microbial oscillator

Angel Goñi-Moreno[1,*] and Martyn Amos[2]

[1] Natural Computing Group, Universidad Politećnica de Madrid, Spain

[2] Department of Computing and Mathematics,
Manchester Metropolitan University, UK

* E-mail: agmoreno@gcn.upm.es, m.amos@mmu.ac.uk

Abstract

Genetic oscillators are a major theme of interest in the emerging field of synthetic biology. Until recently, most work has been carried out using intra-cellular oscillators, but this approach restricts the broader applicability of such systems. Motivated by a desire to develop large-scale, spatially-distributed cell-based computational systems, we present an initial design for a *population-level* oscillator which uses three different bacterial strains. Our system is based on the client-server model familiar to computer science, and uses quorum sensing for communication between nodes. We present the results of extensive *in silico* simulation tests, which confirm that our design is both feasible and robust.

1 Introduction

The growing field of synthetic biology [5, 25, 28] has the potential to impact on many pressing areas of concern, such as health [20, 26], energy [17] and the environment [27]. By engineering bacteria (and sometimes other types of cell), practitioners in the field hope to take advantage of their inherent "biological nanotechnology". This engineering is generally achieved by modifying the natural transcriptional mechanisms and regulatory activities of the bacterium of interest. Collections of bacterial cells have recently been successfully engineered to perform simple tasks, such as emulating light-sensitive film [19], or generating simple patterns [4, 29]. By harnessing and controlling communication and synchronization mechanisms found in such systems, we hope to engineer scalable, robust, fault-tolerant bacterial devices.

Our objective is to design a multi-strain bacterial community with autonomous behaviour. We model our system on the "client-server" architecture familiar to computer science [6], with a single central server and two clients (one "red" and the other "green"). The task we define is that of *oscillation*; by engineering feedback between three different strains, web obtain indefinite switching between "red" and "green" outputs.

In this paper we first briefly review previous work on engineered cellular oscillators. We then describe the architecture of the two clients and the single server strain, which all use standard genetic parts. We describe *in silico* component testing results, before demonstrating, using extensive simulation studies, the feasibility of engineering multi-strain, *population-based oscillators*. Our results suggest that such distributed computations may become more common as the field of synthetic biology matures.

2 Previous work

Synthetic biology is an emerging scientific discipline largely concerned with the engineering of biological systems. The goals of synthetic biology include the "optimization of existing biological systems for human purposes, and the development and application of rational engineering principles to the design and construction of biological systems. A significant amount of work on synthetic biology has concerned *switches* [15] and *oscillators*; here, we focus on the latter.

In physics, an oscillator is a system that produces a regular, periodic "output". Familiar examples include a pendulum or a vibrating string. Linking several oscillators together in some way gives rise to *synchrony* – for example, heart cells repeatedly firing in unison, or millions of fireflies blinking on and off, seemingly as one [31].

Although synthetic genetic oscillators date back to the early 1960s [16], these so-called *Goodwin oscillators* were limited to single genes. The archetype of the *multi-gene* oscillator is known as the *repressilator*, which is a "ring" of genes, each repressing its successor [12, 22]. A detailed discussion of synthetic genetic oscillators is beyond the scope of this paper, but we refer the reader to a recent extensive survey [24].

Recently, genetic clocks have been coupled to produce synchronised oscillations at the level of a cell *population* [9]. Following on from earlier theoretical work [14, 21], this paper demonstrated the feasibility of engineering population-level oscillations. However, the population used was *homogenous*. In nature, there exist bacterial communities known as *biofilms* [10], in which hundreds of bacterial species form a robust and stable community through signalling and co-operation. If the potential of synthetic biology is to be fully realised, we believe that it is important to understand how to engineer communication in *mixed* groups of cells. We therefore describe a scheme for obtaining population-level oscillations using a *three-component* population.

3 A multi-strain bacterial oscillator

In this Section we describe in detail the structure of our population-based client-server oscillator. We show how a three-strain community can, in principle achieve oscillatory behaviour (switching from red to green light output) in an autonomous, synchronous fashion. The basic architecture of our system, depicted in Figure 1, is based on the "client-server" principle of modern computing, in which distributed *client* nodes communicate with a central *server*. In our system, we have one server strain and two client strains. We extend the analogy by considering the role of a *buffer*, which is the nutrient solution in which the cells live and grow (in computing, a buffer is a region of memory in which temporary data are stored). Signals are transmitted between client and server via this "shared memory", through the actions of sensing and deposition. Each cell ("processor") also has its own private internal "memory", corresponding to the space inside the membrane where local functions are performed. As each bacterium is, in effect, an independent processor, the success of our design relies on our ability to make all processors react *simultaneously* to external signals.

Quorum-sensing (QS) [2] has already been studied extensively in the context of synthetic biology [1, 3, 14]. This mechanism facilitates inter-bacterial communication via the generation and receiving of signal molecules [13]. Most importantly, it enables a community-level response to emerge once a certain *concentration threshold* has been reached. It is this mechanism that we will use as the basis of the current study. In what follows, there exist only four different signals, labelled AHLg, AHLr, AHLs and AHLs' (the precise definitions of each will be made clear later). Each cell/processor reacts not to the *absence* or *presence* of a specific AHL signal, but to the signal *level*, or concentration. For that purpose, a *threshold*, δ, for input responses is defined in each cell. If the output, $O(B_i)$ produced by cell B_i is required for activation of B_j, and some signal level is denoted by $|x|$, we assert that when $|O(B_i)| \geq \delta B_j$ then cell B_j is activated. In this way, our model attempts to address one of the biggest problems inherent to single-cell circuits; stochastic expression noise due to inappropriate concentrations of signalling molecules.

In Figure 1, we show the server and two clients; the server is *activated* by both AHLs and AHLs' (producing either AHLr or AHLg respectively); the *green* client is activated by AHLg, producing AHLs and green fluorescent protein, and the *red* client is activated by AHLr, producing AHLs' and red fluorescent protein. We can therefore see how this machine lies dormant until either AHLs or AHLs' is added to the nutrient, after which the system enters a period of oscillation (either red-green-red-... or green-red-green-... respectively). This is achieved by the server cells switching "turns" between red and green client cells. The intended behaviour of the server is shown in Table 1; the important thing to note is that it acts as an XOR (exclusive OR) function, since it is only active when receiving a single input (i.e., when *one* of the clients is active). If, for some reason, either *both* or *none* of the clients are active, then the server should be *inactive* - this is central to the correct operation of the system.

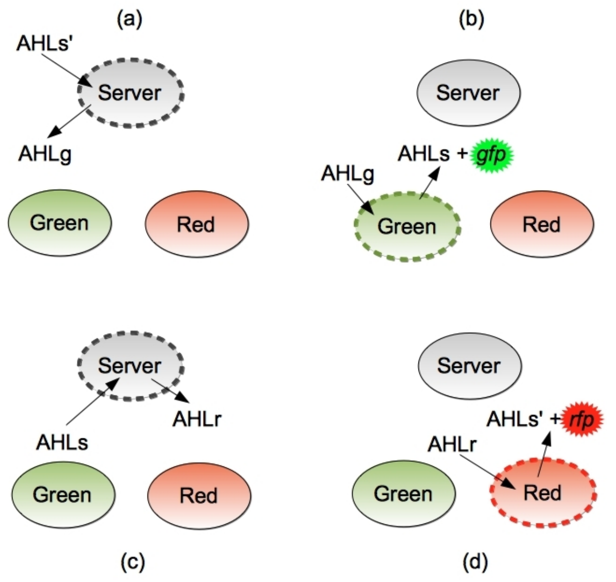


Figure 1: Overall system architecture. (a) Server activated by AHLs'. (b) Green client activated. (c) Server re-activated by AHLs. (d) Red client activated.

We now describe in detail the internal structure of the client and server *E. coli* strains. In order to do so, we first specify the AHL molecules corresponding to the various signals within the system. These are listed in Table 2.

The four quorum sensing systems were chosen carefully, considering (1) sensitivity, (2) bacterial class, and (3) potential conflicts. In [23] the four systems are grouped together in terms of their sensitivity, in [18] the systems are all characterised as being present in particular divisions of specific Proteobacteria, and in [30] it is established that there exist no conflicts between the molecules sensed by the different systems. We therefore strongly believe that the systems we have chosen are appropriate.

AHLs	AHLs'	AHLg	AHLr	S
0	0	0	0	0
0	1	1	0	1
1	0	0	1	1
1	1	0	0	0

Table 1: Truth table for server strain.

Signal	System	Molecule
AHLg	LasI/R	3OC12AHL
AHLr	Rh1I/R	C4AHL
AHLs	LuxI/R	3OC6AHL
AHLs'	SinI/R	3OC14AHL

Table 2: Specific molecules corresponding to signals.

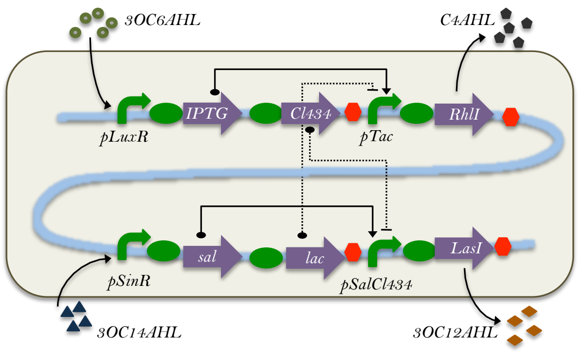


Figure 2: Server cell internal architecture.

3.1 Server cells

The server cells lie at the heart of the system, as they are responsible for implementing the core switching behaviour. In order to implement this, we use two *hybrid promoters*. These are promoters that are regulated by *two* inputs (one inducer and one repressor), and careful design allows them to be combined. In order to activate the *red* output signal from the server, it first requires an input of AHLs.

The detailed structure of the server is depicted in Figure 2. When a server bacterium detects via its membrane that the concentration of AHL$_s$ molecules exceeds the input threshold for that QS system, the inducible promoter *pLuxRs* is activated. As a result of this, the two downstream structural genes are expressed. The production of IPTG molecules is used to stimulate a positive action in the *hybrid promoter* $pTac$ [11] which, in turn, manages the expression of AHL$_r$ molecules by using the gene *luxIr*. At the same time, the expression product of the second gene, *CI434*, represses the hybrid promoter *pSalCI434*[1], so the production of AHL$_g$ is no longer possible due to the inhibitionof *luxIg*. This general subsystem design is duplicated in the server to react symmetrically to each of the inputs the server may receive: AHL$_s$ and AHL$_{s'}$.

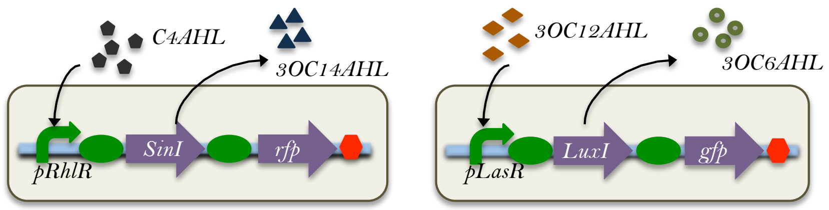


Figure 3: Client cell internal architecture (Left: red, Right: green).

3.2 Client cells

The detailed structures of the client cells are shown in Figure 3. These cells have a much simpler design, because of the lack of synchronization requirements on clients within this model. In the case of the green client, when it senses a sufficient concentration of AHL$_g$ molecules in the environment to raise the threshold of the corresponding QS system, it activates the internal pathway that concludes with the expression of AHL$_s$ molecules and the reporter *(*gpf). The first stage of this is the activation of the inducible promoter *pLuxRg* which allows the transcription and translation of the genes *LuxI* and *gfp*. *LuxIs* is used to produce end-turn signals (in this case, AHL$_s$), which are placed in the shared memory in order to notify the server that the green light that corresponds to half oscillation cycle has been satisfactorily expressed. The design of the red client is exactly the same, only with $SinI$ replacing $LuxI$, and rfp being produced instead of gfp.

4 Experimental results

We now describe the results of simulation-based experiments to investigate the behaviour of both the individual components, and the client-server system as a whole. In order to achieve this, we run two sets of simulations, at different levels of detail. The first (micro-level) set investigates *intra-cellular* behaviour (i.e., at the internal level of gene regulation), and the second (macro-level) set of experiments assesses the effects of *inter-cellular* interactions.

4.1 Component testing: single-cell experiments

For single component testing, simulations are performed using the Tinkercell [7] CAD tool to model the genetic networks of both the clients and the server. This simulation environment provides a framework for the study of the dynamical behaviour of genetic circuits. The user specifies the different components of the circuit (either taken from a standard library or defined on an *ad hoc basis*) and their connectivity, as well as external inputs and outputs. The key detail lies in the correct specification of the nature of connections between components.

[1]Produced by the PKU Beijing team for IGEM2009, http://2009.igem.org

These connections are represented by ODEs, which draw their terms from the individual components. Tinkercell then solves the set of equations on behalf of the user, producing dynamical time-series plots. The Tinkercell schematics for the server and clients are depicted in Figure 4. The server hybrid promoters (c2 and c3 in figure 4) have two different operator regions where the molecules can bind, one for inducible forces and one for repressible forces. When AHLs exceeds the membrane threshold it binds molecules $m1$ (which are always present in the cell) , and creates a_AHLs (active AHLs molecules). Those molecules induce promoter $i1$ activity. The same principle applies for AHLs2 (i.e., AHLs', the labelling is simply an restriction of the software).

Tinkercell provides default equations by which various standard components are connected, and no changes were made to any of these. All concentrations are measured in μM units. However, it is relevant to remark that the strength of the RBSs and the promoters has been increased to a very high value in order to detect possible irregularities in the system (RBS_strength = 20; Promoter_strength = 4). That is why in this single component specification, we observe high values for output responses. In this way, we artificially amplify both "good" and '"bad" signals, in order to easily detect aberrant behaviour. However, this does not *change* in any way the overall behaviour of the system. All the intermediate transcription factors of the system are set up to 0 μM at the beginning of the simulation, and we include a degradation reaction for each.

We now show the results of single-cell simulations for clients and server. We vary the inputs to each in order to confirm that they only produce a threshold output at the correct concentration levels. Figure 5 shows the behaviour of the two client strains for different concentrations of input molecules. Molecule C4AHL, which is the red-specific turn signal, induces the expression of ClientR output. It is very important to note that the concentration of those molecules which raise thresholds is denoted by a value of zero for C4AHL. That is the moment in which ClientR's turn begins, by launching the cascade of its quorum sensing system to produce 3OC14AHL (end turn signals) and red fluorescent proteins (for system output). When C4AHL molecules exceed a concrete saturation level (higher than the threshold), the production of ClientR's output reaches its maximum and the client reaches an steady state. We note that ClientR's operative machinery is completely indifferent to green-specific turn signals (3OC12AHL). A similar behaviour affects the green client strain with opposite molecular perception.

We now describe the simulation tests of the server component of the model. Since noise-attenuation inside the server is one of the most important features of the model, this component is tested extensively with a wide range of values for the degradation rates of the four transcription factors connected to the hybrid promoters. The overall objective is to ensure that the server only ever emits a single signal during any one '"turn". We first performed a set of runs using random values for each degradation rate, which confirmed that the system performed correctly and robustly. We were only able to obtain incorrect behaviour for the system by using abnormal and unrealistic degradation rates. In what follows, we use the default degradation rate (-1) for each protein.

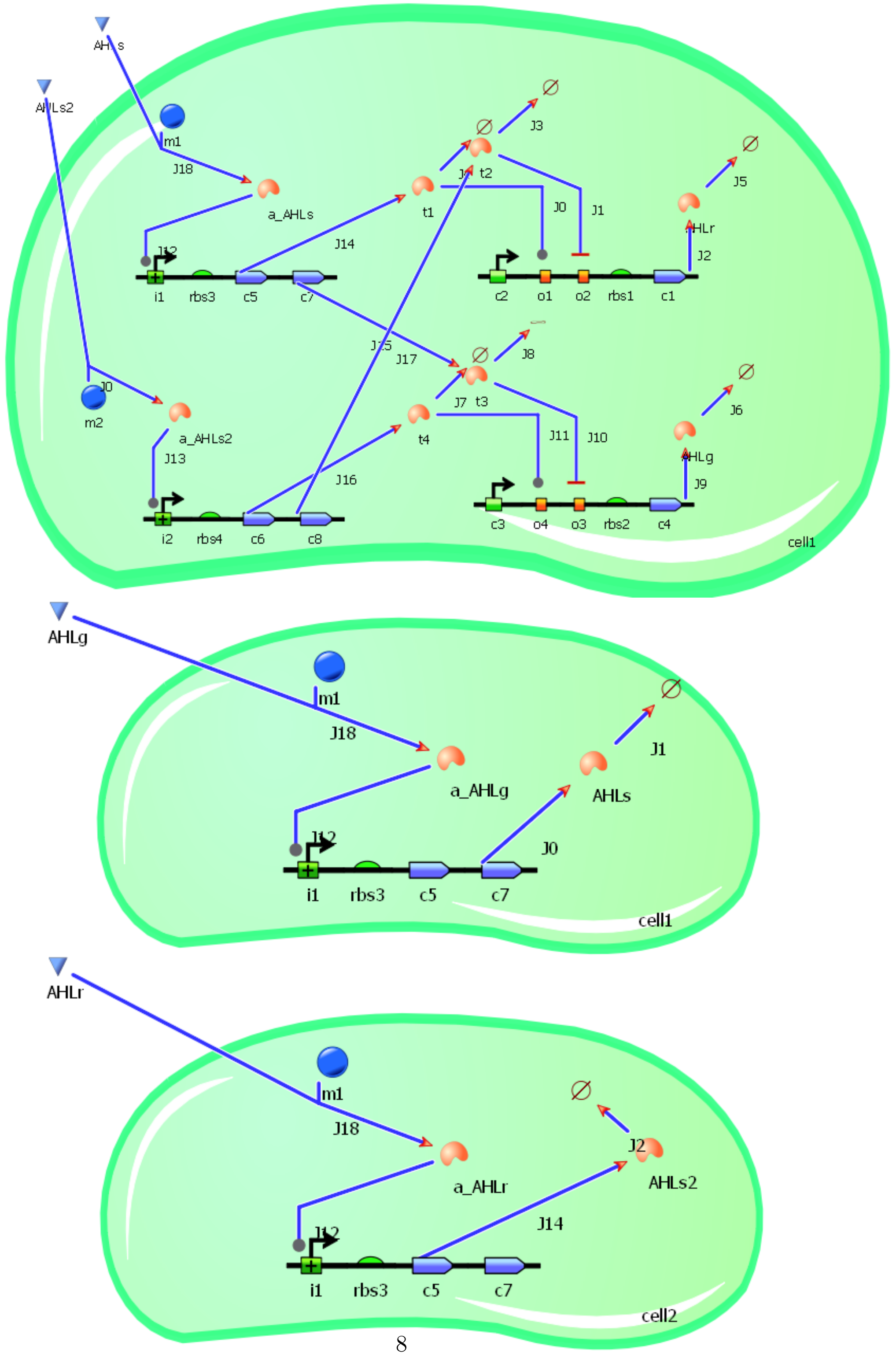
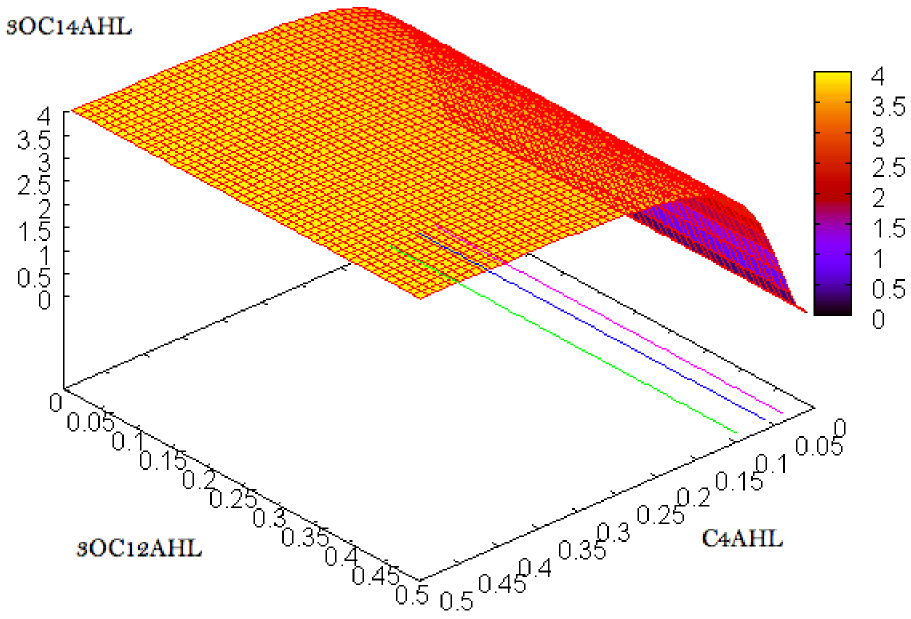


Figure 4: Tinkercell schematics. Top: Server. Middle: Green client. Bottom: Red client.



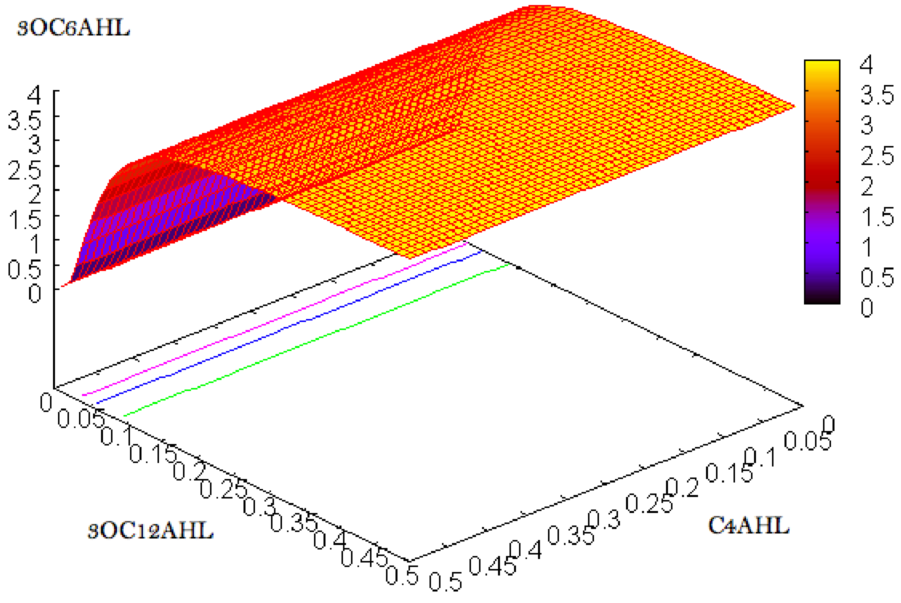


Figure 5: Client behaviour. Top: Red client. Bottom: Green client.

Again, we vary the concentrations of both inputs in order to confirm that the server yields the correct output. The graphs in Figure 6 illustrate the operation of the server during each "turn". Figure 6 (top) depicts the situation in which the server receives red-specific *end turn* signals (3OC14AHL) and produces green-specific turn signals (3OC12AHL). In order to make this graph clear, two important features must be highlighted. Firstly, a zero value for an input signal (x and y axes) represents, as in the client graphs, the instant at which the concentration of the corresponding molecule in the nutrient solution is sufficient to exceed the membrane threshold so that the server can start. However, the most relevant attribute of the server is the interaction of red and green end turn signals inside the cell. As we observe in graph 6 (top), red-specific end turn signals activate the expression of the green turn *if and only if* green-specific end turn signals (3OC6AHL) are *not present* inside the server (or at a very low concentration). This characteristic aids noise attenuation and, therefore, the correct operation of the model in case of saturation of the system, by causing a halt state.

These simulation tests confirm, in principle, the correct functioning of the individual system components. We now describe the results of full-system simulations to assess the overall behaviour of our client-server model.

4.2 System-level experiments

Having confirmed the correctness of the individual component designs, we now seek to investigate the overall system behaviour. In order to achieve this, we treat each of the three components as a "black box" within a simulation based on Michaelis-Menten kinetics [8] and performed using the BioBrick library for Simulink (cite) . The fundamental schematics for each component are the same as in the Tinkercell simulation; the only difference lies in the manner in which the system equations are solved by the software. In both of them, abiological processes are represented by a set of ODEs:

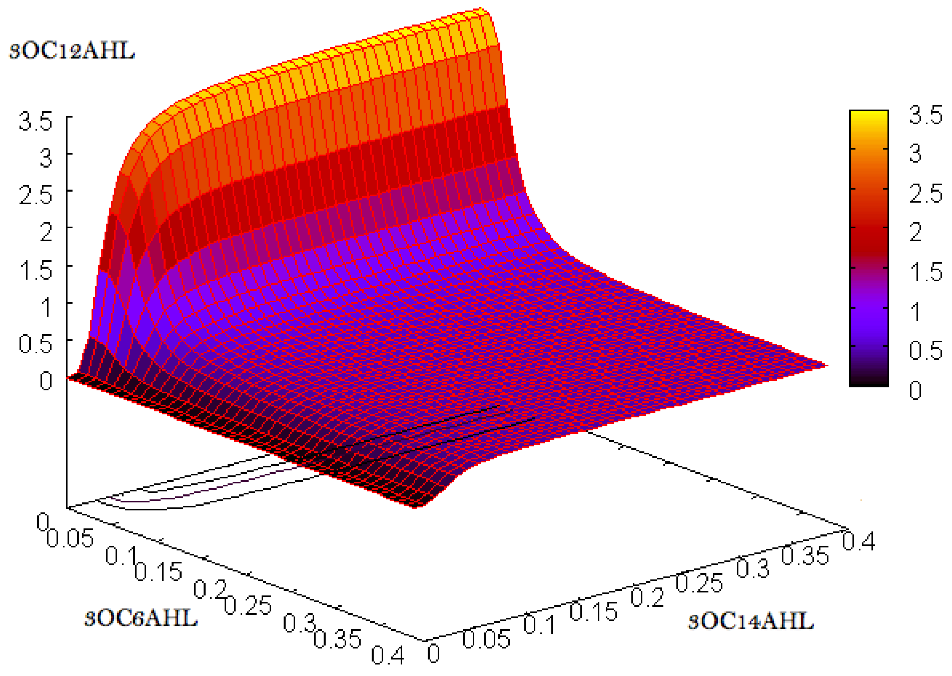
$$\mu(t) = p(t) - \delta\mu(t) \qquad (1)$$

$$p(t) = \beta\frac{FT}{\gamma + FT} + \alpha \qquad (2)$$

$$\eta(t) = \lambda\mu(t) - \delta_p\eta(t) \qquad (3)$$

$$p(t) = \beta\frac{1}{\gamma + FT} + \alpha \qquad (4)$$

Equation 1 describes the genetic transcription process $\mu(t)$, where $p(t)$ is the production rate and δ the linear degradation rate. Equation 2 represents the action of an inducible promoter (needed for the QS system), where FT is the transcription factor, γ a saturation constant, β the maximum transcription rate and α a default transcription constant. For concluding protein production, the model obeys equation 3 where λ is a constant and δ_p the degradation rate of the final output $\eta(t)$. Finally, equation 4 denotes a repressible promoter, which is used to build in Simulink the negative operator of the hybrid promoters.



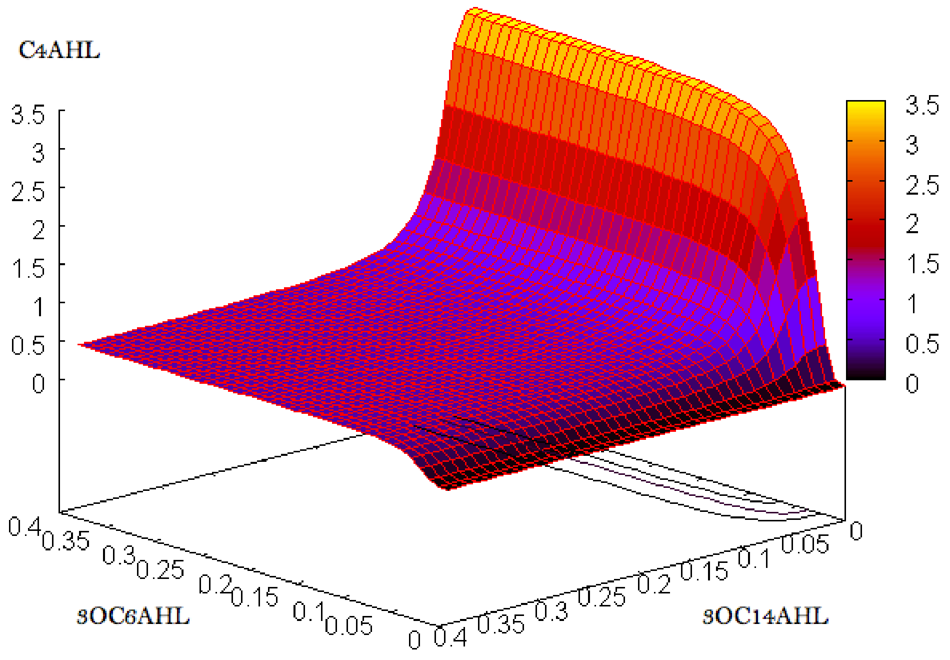


Figure 6: Server behaviour. Top: Green turn output. Bottom: Red turn output.

While Tinkercell uses an integrator (CVODE) to build the final system, the BioBrick library eliminates nonlinearities by linearising the ODEs using a frequency (Laplacian) domain.

We perform three sets of experiments to assess the system-level behaviour. In the first, we assume an idealised situation, with zero cell growth. The second set of experiments investigates a rather more realistic situation, which accounts for cell growth. The third set of experiments simulates the situation in which one client has a different signal degradation rate to the other (in order to both confirm correct system behaviour, and to open up the possibility of different oscillatory patterns).

Before describing the experiments, we first make explicit some assumptions. Given that the minimal concentration (c_m) of molecules that a strain can produce is equal to 0, and the maximal concentration (c_M) is equal to a predefined value (5 in the selected simulations), concentration values within the simulation lie in the interval [0...5]. (percentage proportion). Within that interval, the numeric value of the membrane threshold can be specified within the interval [$\simeq 0.7$ ... $\simeq 3.7$] which represents 60% of the global molecule production (or 2% if repression forces are not considered). In this way, we hope to obtain robust system behaviour across a range of environmental conditions. In what follows, time units are dimensionless, as this parameter is dependent on the concentration value. We note that, in all of the graphs, the level of green fluorescence is coupled to the concentration of AHLs, and the level of red fluorescence is coupled to the concentration of AHLs'. We now investigate three different scenarios: (1) an idealised situation, with a static population, (2) a more realistic situation, with a *growing* population, and (3) the situation where clients have different degradation rates associated with them.

4.2.1 Scenario 1: Static population

We first study the simple situation in which the bacterial population is static (i.e., there is no growth). This is an idealised example, in which we effectively simulate a system containing a single server, and one of each client. The results are depicted in Figure 7.

We observe the expected pattern of oscillation, starting with the red client. This turn is denoted by a rise in concentration of AHLr. The important thing to note is that the concentration of AHLg is basal until the end of the red turn is signalled by a rise in concentration of AHLs'. This triggers the green client's turn, signalled by a rise in concentration of AHLg, the end of which is indicated by a rise in concentration of AHLs. This last rise then triggers the activity of the red client once more, and the cycle continues.

4.2.2 Scenario 2: Growing population

The second experiment considers the situation in the different strain populations observe the usual growth patterns. Rather than explicitly simulating each bacterium or coarse-grained sub-population (which would require a spatially-

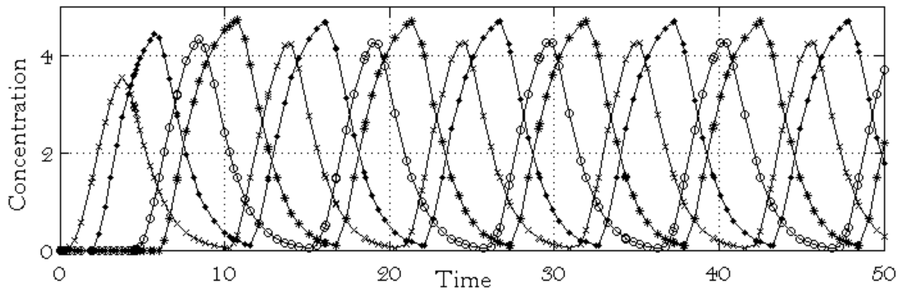


Figure 7: Simulation results, static population. Key: AHLr($\times$), AHLg($\circ$), AHLs'($\bullet$), AHLs($*$).

explicit model, which is beyond the scope of the current study), we reproduce the effect of a growing strain by multiplying the output of a single population member according to the standard log phase growth function.

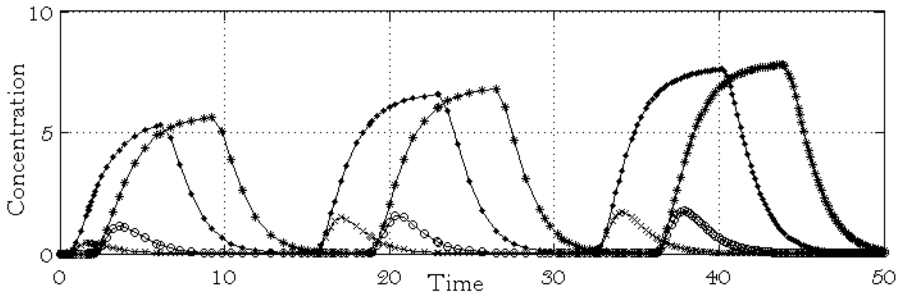


Figure 8: Simulation results, growing population. Key: AHLr($\times$), AHLg($\circ$), AHLs'($\bullet$), AHLs($*$).

Again, we observe the expected system behaviour, but see a gradual rise in the overall maximum concentrations of AHLs and AHLs', which is consistent with population growth. The concentration levels of AHLg and AHLr are lower than in the previous experiment because they correspond to “start” signals to the clients. The server therefore has the capability of ”pausing” the oscillation until the correct conditions are in place.

4.2.3 Scenario 3: Differential client behaviour

In the final experiment we investigate the effect of differential behaviour of the client strains. We modify the degradation rate of the “red off” signal AHLs', so that it is removed much more slowly from the system. In Figure 9 we observe the effect of this change. The red client finishes its turn, but the end-turn molecules degrade more slowly, and are therefore still present in the shared memory. The server notices this red end-turn, and repeatedly yields the turn to the green client until the AHLs' molecules disappear. The system can therefore adapt its behaviour to this new situation. We do not observe a green-red-green-red pattern, but instead see red-green-green-green-green-red.... Importantly, the system dynamically reconfigures the oscillation pattern in a manner that is completely consistent with correct architectural behaviour.

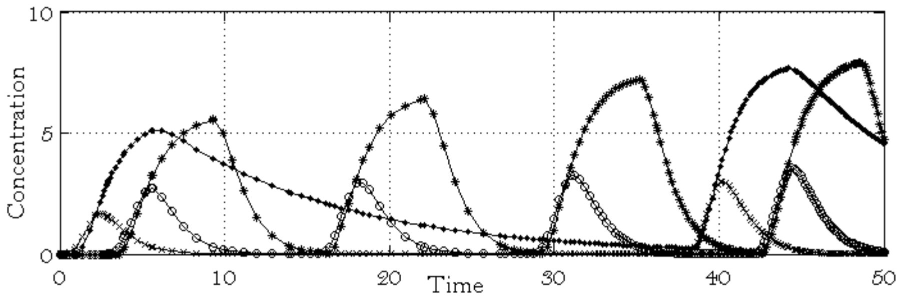


Figure 9: Simulation results, differential client behaviour. Key: AHLr(×), AHLg(○), AHLs'(●), AHLs(∗).

5 Conclusions

In this paper we presented a design for a population-based cellular oscillator, which uses quorum sensing-based signalling within a client-server model. Simulation studies of our design suggest that it is realistic and robust to fluctuations in environmental conditions. Such systems will become increasingly important for synthetic biology, as the field seeks applications in (for example) distributed bio-sensing or tissue engineering. Future work will focus on refinements of the model, as well as its experimental validation.

References

[1] Ernesto Andrianantoandro, Subhayu Basu, David K. Karig, and Ron Weiss. Synthetic biology: new engineering rules for an emerging discipline. *Mol. Syst. Biol.*, 2:2006.0028, 2006. doi:10.1038/msb4100073.

[2] S. Atkinson and P. Williams. Quorum sensing and social networking in the microbial world. *Journal of The Royal Society Interface*, 6(40):959, 2009.

[3] Frederick K. Balagaddé, Hao Song, Jun Ozaki, Cynthia H. Collins, Matthew Barnet, Frances H. Arnold, Stephen R. Quake, and Lingchong

You. A synthetic escherichia coli predator-prey ecosystem. *Mol. Syst. Biol.*, 4:187, 2008. doi:10.1038/msb.2008.24.

[4] Subhayu Basu, Yoram Gerchman, Cynthia H. Collins, Frances H. Arnold, and Ron Weiss. A synthetic multicellular system for programmed pattern formation. *Nature*, 434(7037):1130–1134, 2005. doi:10.1038/nature03461.

[5] Steven A. Benner and A. Michael Sismour. Synthetic biology. *Nat. Rev. Genet.*, 6(7):533–543, 2005. doi:10.1038/nrg1637.

[6] A. Berson. *Client/server architecture.* McGraw-Hill, Inc. New York, NY, USA, 1996.

[7] Deepak Chandran, Frank T. Bergmann, and Herbert M. Sauro. Tinkercell: modular CAD tool for synthetic biology. *J. Biol. Eng.*, 3(19), 2009. doi:10.1186/1754-1611-3-19.

[8] A. Cornish-Bowden. *Fundamentals of enzyme kinetics.* Portland Press London, 1995.

[9] Tal Danino, Octavio Mondragón-Palomino, Lev Tsimring, and Jeff Hasty. A synchronized quorum of genetic clocks. *Nature*, 463(7279):326–330, 2010. doi:10.1038/nature08753.

[10] D.G. Davies, M.R. Parsek, J.P. Pearson, B.H. Iglewski, J.W. Costerton, and E.P. Greenberg. The involvement of cell-to-cell signals in the development of a bacterial biofilm. *Science*, 280(5361):295, 1998.

[11] H.A. De Boer, L.J. Comstock, and M. Vasser. The tac promoter: a functional hybrid derived from the trp and lac promoters. *Proceedings of the National Academy of Sciences of the United States of America*, 80(1):21, 1983.

[12] M. B. Elowitz and S. Leibler. A synthetic oscillatory network of transcriptional regulators. *Nature*, 403(6767):335–3–38, 2000. doi:10.1038/35002125.

[13] W.C. Fuqua, S.C. Winans, and E.P. Greenberg. Quorum sensing in bacteria: the LuxR-LuxI family of cell density-responsive transcriptional regulators. *Journal of Bacteriology*, 176(2):269, 1994.

[14] Jordi Garcia-Ojalvo, Michael B. Elowitz, and Steven H. Strogatz. Modeling a synthetic multicellular clock: repressilators coupled by quorum sensing. *Proc. Natl. Acad. Sci. USA*, 101(30):10955–10960, 2004. doi:10.1073/pnas.0307095101.

[15] T. S. Gardner, C. R. Cantor, and J. J. Collins. Construction of a genetic toggle switch in escherichia coli. *Nature*, 403:339–342, 2000.

[16] B.C. Goodwin. *Temporal organization in cells. A dynamic theory of cellular control processes.* Academic Press London, 1963.

[17] Sung Kuk Lee, Howard Chou, Timothy S. Ham, Taek Soon Lee, and Jay D. Keasling. Metabolic engineering of microorganisms for biofuels production: from bugs to synthetic biology to fuels. *Curr. Opin. Biotechnol.*, 19(6):556–663, 2008. doi:10.1016/j.copbio.2008.10.014.

[18] E. Lerat and N.A. Moran. The evolutionary history of quorum-sensing systems in bacteria. *Molecular Biology and Evolution*, 21(5):903, 2004.

[19] Anselm Levskaya, Aaron A. Chevalier, Jeffrey J. Tabor, Zachary Booth Simpson, Laura A. Lavery, Matthew Levy, Eric A. Davidson, Alexander Scouras, Andrew D. Ellington, Edward M. Marcotte, and Christopher A. Voigt. Synthetic biology: engineering *escherichia coli* to see light. *Nature*, 438(7067):441–442, 2005. doi:10.1038/nature04405.

[20] Timothy K. Lu and James J. Collins. Engineered bacteriophage targeting gene networks as adjuvants for antibiotic therapy. *Proc. Natl. Acad. Sci. USA*, 106(12):4629–4634, 2009. doi:10.1073/pnas.0800442106.

[21] D. McMillen, N. Kopell, J. Hasty, and JJ Collins. Synchronizing genetic relaxation oscillators by intercell signaling. *Proceedings of the National Academy of Sciences of the United States of America*, 99(2):679, 2002.

[22] S. Müller, J. Hofbauer, L. Endler, C. Flamm, S. Widder, and P. Schuster. A generalized model of the repressilator. *Journal of Mathematical Biology*, 53(6):905–937, 2006.

[23] A. Pai and L. You. Optimal tuning of bacterial sensing potential. *Molecular Systems Biology*, 5(1), 2009.

[24] O. Purcell, N.J. Savery, C.S. Grierson, and M. di Bernardo. A comparative analysis of synthetic genetic oscillators. *Journal of The Royal Society Interface*, 2010. doi:10.1098/rsif.2010.0183.

[25] Priscilla E. M. Purnick and Ron Weiss. The second wave of synthetic biology: from modules to systems. *Nat. Rev. Mol. Cell Biol.*, 10(6):410–422, 2009. doi:10.1038/nrm2698.

[26] Dae-Kyun K. Ro, Eric M. Paradise, Mario Ouellet, Karl J. Fisher, Karyn L. Newman, John M. Ndungu, Kimberly A. Ho, Rachel A. Eachus, Timothy S. Ham, James Kirby, Michelle C. Y. Chang, Sydnor T. Withers, Yoichiro Shiba, Richmond Sarpong, and Jay D. Keasling. Production of the antimalarial drug precursor artemisinic acid in engineered yeast. *Nature*, 440(7086):940–943, 2006. doi:10.1038/nature04640.

[27] Gary S. Sayler, Michael L. Simpson, and Chris D. Cox. Emerging foundations: nano-engineering and bio-microelectronics for environmental biotechnology. *Curr. Opin. Microbio.l*, 7(3):267–273, 2004. doi:10.1016/j.mib.2004.04.003.

[28] Louis Serrano. Synthetic biology: promises and challenges. *Molecular Systems Biology*, 3, 2007. doi:10.1038/msb4100202.

[29] Takayuki Sohka, Richard A. Heins, and Marc Ostermeier. Morphogen-defined patterning of *escherichia coli* enabled by an externally tunable band-pass filter. *J. Biol. Eng.*, 3(10), 2009. doi:10.1186/1754-1611-3-10.

[30] L. Steindler and V. Venturi. Detection of quorum-sensing N-acyl homoserine lactone signal molecules by bacterial biosensors. *FEMS Microbiology Letters*, 266(1):1–9, 2007.

[31] Steven Strogatz. *Sync: The Emerging Science of Spontaneous Order*. Penguin, 2003.