Engineered gene circuits capable of reinforcement learning allow bacteria to master gameplaying

Adrian Racovita[1]†, Satya Prakash[2]†, Clenira Varela[2], Mark Walsh[2], Roberto Galizi[3], Mark Isalan[4], Alfonso Jaramillo[1,2]*

[1]*De novo* Synthetic Biology Lab, I2SysBio, CSIC-University of Valencia, Paterna, Spain.

[2]School of Life Sciences, University of Warwick, Coventry, UK.

[3]Centre for Applied Entomology and Parasitology, School of Life Sciences, Keele University, Keele, UK.

[4]Department of Life Sciences, Imperial College London, London, UK.

*Corresponding author. Email: Alfonso.Jaramillo@synth-bio.org

†These authors contributed equally to this work

Abstract:

The engineering of living cells able to learn by themselves algorithms such as playing board games —a classic challenge for artificial intelligence— will allow complex ecosystems and tissues to be chemically reprogrammed to learn complex decisions. However, current engineered gene circuits encoding decision-making algorithms have failed to implement self-programmability and they require supervised tuning. We show a strategy for engineering gene circuits to rewire themselves by reinforcement learning. We created a scalable general-purpose library of *Escherichia coli* strains encoding elementary adaptive genetic systems capable of persistently adjusting their relative levels of expression according to their previous behavior. Our strains can learn the mastery of 3x3 board games such as tic-tac-toe from a *tabula rasa* state of complete ignorance. We provide a general genetic mechanism for the autonomous learning of decisions in changeable environments.

One-Sentence Summary:

We propose a scalable strategy to engineer gene circuits capable of autonomously learning decision-making in complex environments.

Animal brains are powerful decision-making devices able to learn autonomously. Computational design of gene circuits has been used to experimentally implement biological adaptive behaviors(*1-8*), but not advanced decision making, which was achieved artificially using physical and chemical systems by engineering memory units with neuromorphic computing capabilities(*9-15*). Engineered gene circuits endow living cells with new decision making skills and their reprogramming could be used to improve the quality of decisions to a given problem. Unfortunately, their reprogramming has focused on modifying the encoding DNA, such as mutating and recombining regulatory regions. This adaptation requires the directed rewiring(*16, 17*) of gene circuits, demanding the precise adjustment of every interaction. This is also the case with the recent reports on the engineering of artificial neural networks (ANN) in bacteria(*14, 18*), where the experimenter infers the needed experimental adjustments through the rational or computational design of weights. This hampers the engineering of large systems where the individual adjustment of parameters would be impractical.

To dramatically simplify the capability to train a gene network towards a complex behavior, irrespective of its size, the computation and implementation of the required adaptation should be encoded in the gene network itself. We therefore propose a genetic strategy where a library of gene circuits encoded in intracellular plasmid mixtures can autonomously adapt their decisions towards a targeted behavior by shifting their plasmid heteroplasmy. We test the capability for learning complex decision-making in living bacteria through the gameplaying of board games, a common benchmark in artificial intelligence(*19*).

Inducible antibiotic resistance genes allow the adaptation of co-encoded genes by shifting plasmid ratios

We designed a stable intracellular plasmid mixture of gene circuits where their expression and mixture fractions are controlled by each other. To create a stable intracellular plasmid mixture, we co-transformed the cells with a mixture of two almost identical multi-copy plasmids, P1 and P2, both maintained with the same ampicillin resistance gene (AmpR). We designed the P1 and P2 plasmids to stabilize their copy number ratios within a cell, by having the same length(*20*) including the fluorescent markers, a common inducible promoter and a translational insulator sequence. P1 and P2 encode inducible operons for fluorescent proteins followed polycistronically by fusions of antibiotic resistance proteins (Kan^R/Cm^R for kanamycin/chloramphenicol resistance) or corresponding non-functional "dead" forms ($dKan^R$/dCm^R): P1, mCherry, Cm^R-$dKan^R$; P2, EGFP, dCm^R-Kan^R (Fig. 1A). P1 and P2 have the same medium-copy replicon. The red/green fluorescence reporter gene mCherry/EGFP and antibiotic resistance proteins are expressed when the promoter is induced with a cognate chemical inducer. We co-transformed the P1 and P2 plasmids into *Escherichia coli* DH10B containing the Marionette cassette (which enables the use of chemically-driven inducible promoters with minimal cross-talk activation)(*21*). We chose 9 inducible promoters from the Marionette cassette to construct a library of P1 and P2 plasmids producing to 9 strains (Fig. 1A and Tables S10-11).

The strains co-transformed with the plasmids P1 and P2 realize a minimal gene circuit, which we call *memregulon* (contraction for *memory regulon*, analogous to the memristor element used in electronic circuits with neuromorphic behavior(*22*)). We denote by a and b the P1 and P2 DNA-copy number in a cell respectively, and we define the fraction of the P1 plasmid in cells co-transformed with both P1 and P2 plasmids ($a/(a+b)$) as *weight* (Fig. 1A), analogously to weights in ANN. We say that a memregulon is active when its promoter is induced. An active memregulon in presence of chloramphenicol or kanamycin readjusts its P1 and P2 DNA levels,

therefore its *weight*, modifying the amount of gene expression of the genes encoded in the plasmid P1 (mCherry in this work, but other genes could be used). We obtained an accurate estimation of a memregulon's weight from fluorescence alone by using distinguishable fluorescent proteins (mCherry and EGFP) in P1 and P2 (see Eq. (2.1) in supplementary text), which we confirmed (Fig. S2) via qPCR (R^2=0.94) and DNA sequencing (R^2=0.99). Fluorescence measurements of *weights* will be referred as *F-weights*. The Cm^R and Kan^R genes enable the cellular antibiotic selection for higher *a* and *b* only if the promoter is active. Note that increasing *b* implies decreasing *a*, because the total plasmid copy number (*a*+*b*) is conserved. Chloramphenicol and kanamycin selection of activated memregulons shift the population distribution of P1:P2 plasmid ratios by increasing the average *a* or *b* respectively. Single-cell *F-weight* measurements of memregulon cultures show that the variation of the average weight is due to the decrease of the cell numbers with highest weights (Fig. S1, Table S18).

The red fluorescence of a memregulon is proportional to the red fluorescence of the corresponding P1-only cells (Fig. 1B and Fig. S16A). For instance, 0.5 weight memregulon strains have 50% of the red fluorescence per cell of the cells transformed only with the plasmid P1 with the same promoter. Traditionally, achieving this control of gene expression would be very challenging because it would require reengineering the mCherry promoter with suitable mutations to weaken the transcription rate. Fig. 1B shows that the 0.5 weight cells have indeed a red fluorescence per cell significantly identical to half of the induced and non-induced values of cells transformed only with the P1 plasmid ($p<0.01$), which contrasts with the inability of transcription regulation to lower the non-induced values. Therefore, a memregulon consists of an inducible gene (the fluorescent reporter in plasmid P1), a memory (the P1:P2 plasmid mixture) and an inducible learning device (the antibiotic resistance cassette paired with its knockouts). The memregulon changes its fluorescence reporter activity by changing its memory only when both chemical inducer and a suitable antibiotic are added.

In the following, we grow and maintain bacterial cultures of different memregulon strains on LB agar plates and measure the weight of each memregulon strain using fluorescence or DNA quantifications, where we always copy cocultures before each measurement via replica plating (Fig. 1C). As cultures are allowed to grow after each replica plating, we asked if this could alter the weights. We show that the *weight* remains constant at the population level for many days and consecutive replica plating procedures (Fig. 1D), effectively functioning as a genetic memory system(*20*).

Memregulons produce gene circuits able to adapt their expression levels by self-modifying their DNA copies

Memregulon weight can be altered by culturing cells with specific antibiotics and promoter inducers. For example, selection with kanamycin or chloramphenicol respectively decreases or increases the *weight,* corresponding to a reduction or increment of mCherry fluorescence levels. The cultures on a parental plate are transferred to a new plate that contains either kanamycin or chloramphenicol, ampicillin and the cognate inducers through a replica plating procedure (Fig. 1E). We stop the antibiotic selection with a subsequent replica plating to an ampicillin-only plate. Only the active memregulons changed their weight significantly ($p<0.01$, Fig. S3).

As the promoters might have had a small crosstalk activation with noncognate inducers, we measured the change in *weight* in the presence of cognate and non-cognate inducers, which showed significant variation in *weight* ($p< 0.01$) only for the cognate inducer (Fig. S4 and S5). This means that different memregulons can be combined and each memregulon can

independently adjust its weight. Using an independent chemical inducer for each memregulon allows each mCherry operon to persistently set its own expression levels by adding kanamycin/chloramphenicol, instead of manually modifying the mCherry expression levels through external manipulations of the plasmid DNA copy number. This enables the local and unsupervised training of *weights*, a desired feature in the training of ANN(*23*).

We can use the combined output (*e.g.* fluorescence) of a set of active memregulons for decision making. For example, if the output is not (or is) desired, we then reproduce the environmental condition, activating memregulons in the presence of kanamycin (or chloramphenicol) to downregulate (or upregulate) their expression, thus contributing to decision making. This allows training by self-programming of the decision-making, by a stepwise downregulation/upregulation of the memregulons' contribution to wrong/correct decisions.

Choosing the highest weight among independent memregulon cocultures allows an experimental reinforcement learning algorithm to find the optimal path in decision trees

The distributed multicellular circuits strategy (DMC)(*24, 25*) allows us to explore whether a coculture of strains, containing memregulons, can learn complex decision-making. For this, we initially challenged the cultures with a mathematical problem equivalent to solving a maze (Fig. 2A), where a "rat" must learn how to find the path to the exit without backtracking. Although this corresponds to one of the simplest decision trees, it allows us to define the methodology to be used for more complex problems. Each path has two crossings with 3 possible diversions each and we identify a path as (x, y) with x and y = 1, 2, or 3. The first encountered crossing (a) is assigned to the inducer L-arabinose (Ara) and the others (b, c and d) to the inducer 3-hydroxytetradecanoyl-homoserine lactone (OHC14), which creates a decision tree of 9 leaves (Fig. 2B). The optimal path follows the diversions x=1 and y=2 at the first and second encountered crossings respectively. We set up three cocultures of two strains at a 1:1 cell ratio. Each coculture contains strains with the pBAD and pCin memregulons of different weights, encoding Marionette promoters(*21*), inducible under the chemicals Ara and OHC14 respectively. The diversion decided on at the level 1 and 2 crossings is defined as the number of the coculture with the highest pBAD and pCin weight. The starting cultures were picked such that they had weights where their initial decisions were the furthest from the optimal path. The weights are measured after replica plating measurement of the red and green fluorescence, adding the chemical inducer designated to the crossing (Fig. 2C). We average the weights of 3 biological replicates. If the decisions (x, y) do not follow the unique correct path, we apply a negative selection to cocultures x and y with kanamycin and the inducers Ara and OHC14 respectively to all biological replicates. Our learning only updates the weights of active memregulons at the position of highest weight. We conduct this cycle of measurement and negative reinforcement twice until no more kanamycin selection is needed because the memregulons have modified their weights to follow the optimal path (Fig. 2D). Controls where the learning is done with either swapped antibiotic or swapped inducers show no change in decisions (Fig. S6).

Generalizing the experimental reinforcement learning algorithm allows finding the optimal strategy in the tic-tac-toe game

Because memregulon weights are maintained stably in cocultures (Fig. S7), we investigated whether we could scale up to use cocultures of more memregulon strains to learn how to master a board game. As done with the early computers, we chose the familiar tic-tac-toe game, a two-player game on a 3x3 board, where the two players ("X" and "O") alternately occupy one vacant board position; the winner is the first player that obtains 3 matching symbols on any row,

column, or diagonal. This game was studied recently using DNA computing(*9, 26*), which required implementing custom 3-input logic gates with catalytic DNA. However, it is not necessary to use combinatorial gates to implement expert players if the decisions are made by choosing the highest weight (called winner-take-all, WTA, strategy) even when using linear positive weights as done here(*27*) (Fig. S8).

It is also useful to define a measure of the general skill level at a game, alike to the Elo ranking(*28*). We use a computer simulation to play all possible games. For this, we input the measured F-weights into a simulation parametrized with our experimental data (supplementary text section 3.1, Table S2), where we evaluate the percentage of won or drawn games (called *expertise*) when playing all possible matches.

As an example of how reinforcement learning can automatically train the weights to achieve a complex computation, we generalized our previous experimental learning algorithm (Fig. 2C) to two-player games. The diversions at each crossing would now correspond to the possible moves at each round (equivalent to the level in the maze). One of the players will be a trainer (player X) and player O will be a bacterial player consisting of a set of cocultures for each of the board positions (excluding the central, played first by player X) (Fig. 3A, left). We arbitrarily assigned a chemical inducer to each of the 9 board positions (Fig. 3A, right). The cells play a match against an opponent by reading their F-weights through replica plating fluorescence measurements (Fig. 3B). The experimental algorithm is as follows (see Fig. 3C): As in the maze example, the chemicals activate the memregulons' promoters involved in a decision vertex, but now the simultaneous use of more than one inducer to measure the F-weights allows assigning most of the time to each vertex the state of the game given by all the opponent's positions. To know O's decision on a move, we induce all cocultures at allowed positions with the inducers assigned to X's played positions. The coculture with the highest "multi-inducer" F-weight (see eq. (2.2) in supplementary text) decides the O's next move. After several rounds, the match finishes and, if the O player loses, we apply a negative reinforcement learning operation to the O cocultures only at the positions occupied by the O player (Fig. 3C). The next matches are played with the updated cocultures.

An example of a match is overviewed in Fig. 3D: After player X starts at the center (round 0), player O could move at any of the other 8 unoccupied positions and, therefore, we consider cocultures at all of them. We do replica F-weight measurements to the cocultures by inducing them with 3-oxohexanoyl-homoserine lactone (OC6, inducer assigned to the center position, where X has moved) and then we choose the position where its coculture had the highest F-weight. In the next round, X makes another move (corresponding to the position assigned to Ara) and we inquire about O's move by inducing the 6 cocultures (at unoccupied positions) with OC6 and Ara (the two positions currently occupied by X), and measuring the highest F-weight among them. O loses in round 3, so we apply a negative reinforcement operation with kanamycin selection (Fig. 1E), in the cocultures at positions previously occupied by O (Fig. 3D, encircled in green), adding all the inducers corresponding to X's moves before round 3 (OC6, Ara, & OHC14), which lead to the losing decisions of O. After this learning, we updated 3 cocultures, which become new parental plates for replica measurements, together with the unchanged 5. Bacteria play new matches until a match ends in a draw and the O player achieved mastery.

A random player of 9-memregulon cocultures learns to master tic-tac-toe by playing using reinforcement learning

We asked if our experimental algorithm could allow a naïve bacterial player O to learn mastering the tic-tac-toe game. We chose bacterial cultures to be second player because the naïve player had a low starting expertise (20%). The starting cocultures (denoted by O_0) consist of the same 9 memregulon strains at equal cell ratios and equal weights at all 8 positions (Fig. 4A, left). This experimentally implements a naïve player in a *tabula rasa* state because all positions have the same cultures and interrogating for the highest weight would give a random position. We performed all the experiments in 3 biological replicates.

O plays a tournament against a trainer player X. The trainer has been computationally designed to provide the fastest learning (see supplementary text section 3.6 and its weights in Table S5). We show in Fig. 4B the F-weights of the cocultures at allowed positions as filled red circles (containing the F-weight value multiplied by 100). We average the F-weight of the 3 biological replicates to compute the highest value, which represents O's decision (O's move in the next round). Player X wins match 1, which leads to a negative reinforcement (L1) of the O_0 cocultures at the 4 positions occupied by O in round 4, to produce O_1. We apply the negative reinforcement to all 3 biological replicates. The weight decreases at those positions and the measurement of O_1 in round 0 shows that the position with the highest F-weight has changed, leading to a new decision after learning (Fig. 4B). We compute the expertise of each biological replicate after each learning. (Fig. 4C). Data file S9 details the computation of the O player's expertise after each learning (as well as for the players of Fig. S9 and S13), by showing the results of using the measured F-weights to play every possible tic-tac-toe match. The O_1 cocultures continue playing (see Fig. 4A, right and bottom) and losing against the trainer (matches 2 to 8), suffering subsequent negative reinforcements (L2 to L8, Fig. 4A right) to each biological replicate, which further changes the cocultures (O_2 to O_8). The expertise did not increase monotonously (it decreased in learning L4, Fig. 4C), but it reached 100% for all replicates in O_8. We also validated experimentally the mastery of O_8 by having the cocultures play one match against 7 expert automatons (Fig. 4B shows an example of this in match 9, see Fig. S10A for the others) and all matches ended in a draw (the best outcome when playing as second player against an expert in tic-tac-toe).

Although the O_8 cultures acquired their mastery by playing 8 games, they have the capability to win arbitrary matches (Fig. 4C, Fig. S10B). As a positive control, we manually created a player using weights with rationally designed values (supplementary text section 3.4) to implement an expert player (O_{RW}) (Fig. S11). A match between cocultures O_{RW} and an expert automaton led to a draw. Two alternative learnings aimed at introducing the smallest variations were performed as negative controls, starting from O_7; using either negative reinforcement with a swapped inducer (O_7^a) or using chloramphenicol instead of kanamycin (O_7^b) did not improve the expertise, as the player lost against the expert automaton (Fig. 4B). We also verified that the cocultures maintained their expertise in time after cold storage (at 4 °C or -80 °C) (Fig. 4D, S12). Reinforcement learning also allowed *tabula rasa* bacterial cocultures to reach mastery when playing as a first player (Fig. S13, its computationally designed trainer is included in Table S4). The experimental details for playing against a trainer are given in Tables S7-8.

Two bacterial players can also learn together by playing against each other. To exemplify this, we set up 2 cocultures of 2 memregulon strains, both chosen to have some knowledge of the game (having X and O expertises of 90% and 48% respectively) and able to achieve mastery in the fewest number of learnings. We performed a tournament of memregulon cocultures playing among themselves and applying reinforcement learning with positive or negative rewards to the

players winning or losing matches. Both cocultures reached mastery after one match (Fig. S9, experimental details in Table S9).

Memregulon cocultures can also learn mastering arbitrary 3x3 board games

To explore the capacity of our cocultures of 9 memregulon strains to learn other 3x3 board games, we performed computer simulations of cocultures of 9 memregulons at every position of a 3x3 board except the center, showing that they can learn in less than 35 reinforcement learnings (supplementary text section 3.3, Fig. S14C) 98% of the possible games in this board (Fig. S14A for the most difficult game of the ones sampled). Moreover, they could even learn how to simultaneously master more than one game at the same time, although there is a limit to how many games can be mastered simultaneously after being learnt in succession (Fig. S14B). In some cases, we found that repeated learning tournaments required enough reinforcement learning steps that some weights vanished (Fig. S15B). If a weight vanishes, the P1 plasmid is lost, and so is the ability of a cell to store a memory, because it is not possible to have a P1 and P2 mixture anymore. To rescue the weight before this occurs, we add to the experimental algorithm an operation that we call *memregulon fusion* (yellow box in Fig. 3C). For this, we mix each memregulon strain culture with another one that contains the same memregulon with a weight of 0.5. This mixture operation changes all weights by averaging each of them with 0.5 (see supplementary methods and Fig. S15A). This averaging increases the weights smaller than 0.5 (Fig. S15C) while maintaining the position with the highest weight, and therefore the player's expertise.

To allow our experimental learning optimization to converge towards mastery of arbitrary games and/or mastering of multiple games simultaneously, we also need to avoid getting non-expert players trapped in draws where no more learning occurs. For this, we further extended the experimental learning algorithm by applying a reinforcement learning using chloramphenicol (instead of kanamycin) for selection. After the last match where a negative reinforcement learning with kanamycin was applied, we incubated the cocultures with chloramphenicol and the inducers used in the match (supplementary methods). We call this reinforcement "unlearning" (yellow box in Fig. 3C), mirroring a similar concept from machine learning(*29*). After one unlearning, the bacteria altered their decisions and therefore their expertise also changed, thus avoiding getting trapped in draws (Fig. S9G).

Discussion

We can better appreciate the computational power of our memregulon cocultures by identifying them with a single-layer linear ANN (or linear perceptron) (*30*) of three 2-input neurons (maze example) or eight 9-input neurons (3x3 board games), with an indirect non-linearity coming from the decision on the highest weight. Such networks can be universal function approximators, even when using positive weights exclusively (*27*). The change of weight only when a memregulon is active is central to learning. This follows Hebb's idea(*31*) that the changes in synaptic strength (weight) should be proportional to the presynaptic cell activity and to a function of the postsynaptic cell activity. Long-term potentiation and long-term depression would correspond to a weight increasing and decreasing respectively(*32*). Moreover, similarly to neuromodulated synaptic plasticity, because their change of weights requires the memregulon activity together with either kanamycin or chloramphenicol, these antibiotics act as neuromodulatory signals(*33*).

Our memregulon cocultures could be generalized to other 2-player 3x3 board games where the pieces are added but never moved or removed (supplementary text section 3.3). In Fig. S14C we

show computationally that the number of learnings required for 95% of possible zero-sum games would be between 5 and 35. This could be implemented experimentally within 23 days, considering that each reinforcement learning takes 15 hours. Larger boards would require cocultures of more than 9 strains where weight stability would have to be validated again. We do not necessarily need more orthogonal inducible promoters because we could rely on combinatorial promoters. Currently, there are few examples of combinatorial promoters and to exemplify this we engineered 4 of these requiring two Marionette inducers at high concentration to achieve activation. Fig. S17 shows examples of 4 memregulons with combinatorial promoters pTet:Ttg, pTac:Tet, pTac:Van and pVan:Ttg. This combinatorial promoter strategy could allow creating cocultures of up to 36 memregulons using the Marionette strain.

Memregulons also allow for the construction of gene circuits with predefined behavior because the red fluorescence per cell linearly correlates with its weight (Fig. S16A). Although positive and reinforcement learning with positive or negative rewards could be thought to be equivalent to positive and negative selections in directed evolution(*34*), here we do not have mutations, which allows for a smoother, faster and reversible traversing of the phenotypic landscape. Memregulons maintained their weight in solid cultures across many days, suggesting the possibility of using them in ecosystem-level gene circuits(*35*). Further developments could involve providing a mechanism to adapt the topology of gene circuits(*36*). genetically encoding the computation of the maximum output among positions(*37*), negative selection markers(*38*), CRISPR to cleave(*20*) or regulate(*39*) the plasmid copies, engineered RNA replicons(*40*), engineered microbial ecosystems(*41*), as well as adding an extra memregulon library to each player, designed to receive the output of the first library through a cell-cell communication system, mimicking a hidden layer in a neural network. This would enable the processing of more complex information and, therefore, learning more advanced algorithms.

Adaptive gene circuits could already exist in prokaryotic or eukaryotic systems as a non-Darwinian adaptation tool(*42*). Heterozygotic mutations in multicopy plasmids(*43*), polyploid Archea(*44*) or in mitochondrial DNA (microheteroplasmy)(*45*) maintain the ratios of wild-type to intra-cellular mutations. As a mutation in a growth-altering gene under a regulation could suffice to set up a reinforcement learning, it may be possible to infer memregulons in nature by identifying a mapping among environmental conditions, genes, inducible promoters, and selection markers with their inactivating mutations. This mapping would establish in fact a language for “teaching” algorithms to these cells. Reinforcement learning with memregulons provides a strategy for the unsupervised adaptation of complex gene circuits with a large, unknown number of interactions, which will allow for the engineering of genetically encoded general-purpose computational devices capable of self-learning, opening the way to the engineering of synthetic living artificial intelligence.

Acknowledgments:

We acknowledge M. Kushwaha, M. Fuegger and T. Nowak for discussions.

Funding:

Ministerio de Ciencia e Innovacion PID2020-118436GB-I00 (AJ)

BBSRC BB/P020615/1 (MI, AJ)

EPSRC-BBSRC grant BB/M017982/1 (AJ)

EU grant 610730 (AJ)

School of Life Sciences departmental allocation, Keele University (RG)

Volkswagen Foundation grant LIFE 93 065 (MI)

Author contributions:

Conceptualization: AR, AJ

Software: AR, AJ

Formal analysis: AR, AJ

Methodology: AR, SP, AJ

Investigation: AR, SP, CV, MW, RG, AJ

Visualization: AR, AJ

Supervision: AJ

Writing – original draft: AR, AJ

Writing – review & editing: AR, SP, CV, MW, RG, MI, AJ

Competing interests:

Authors declare that they have no competing interests.

Data and materials availability:

All data are available in the main text or the supplementary materials.

Supplementary Materials

Materials and Methods

Supplementary Text

Figs. S1 to S18

Tables S1 to S20

References (*1–19*)

Data S1 to S10

Figure 1

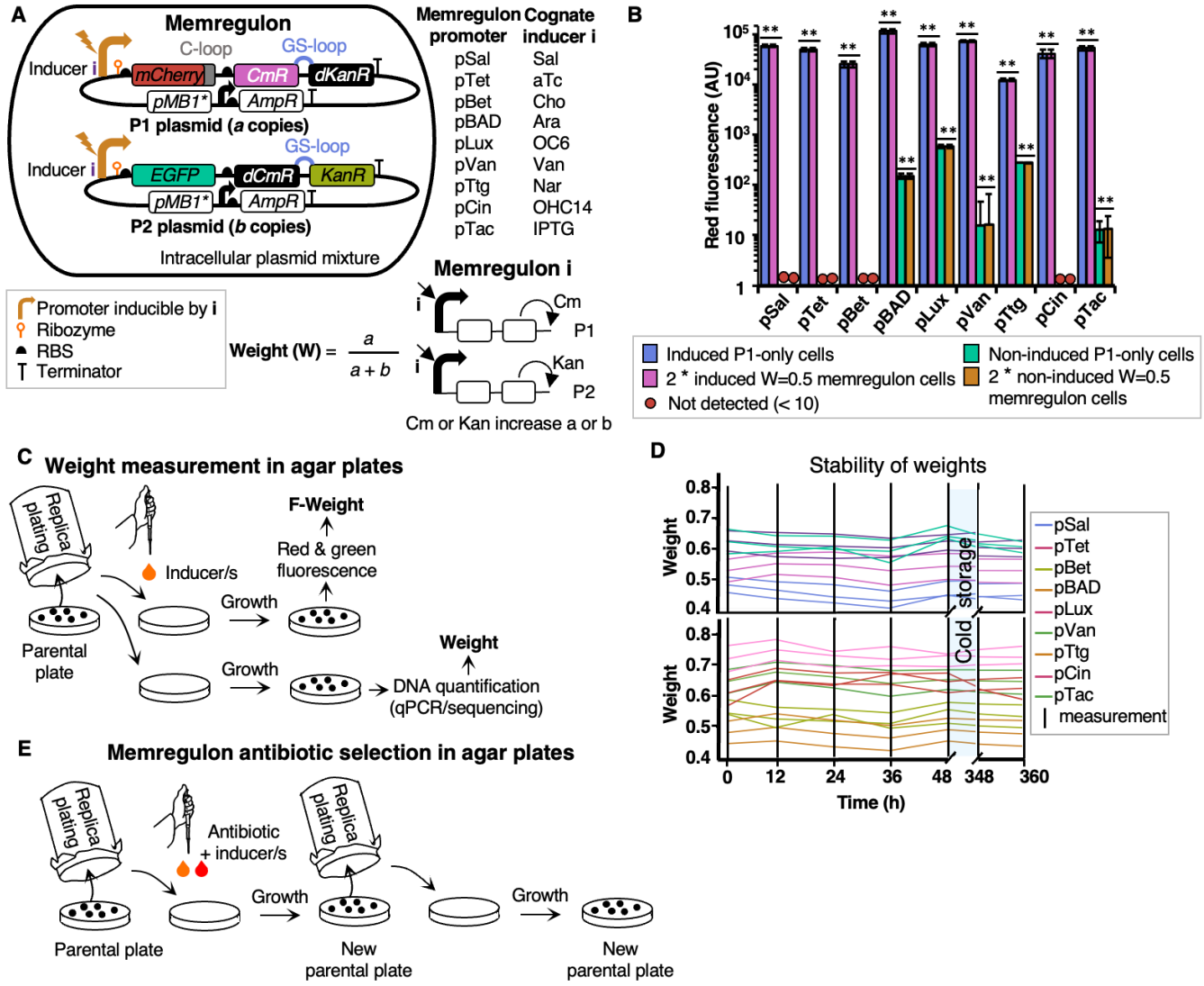
A
Memregulon
C-loop
GS-loop
Inducer i
mCherry
CmR
dKanR
pMB1*
AmpR
P1 plasmid (a copies)
EGFP
dCmR
KanR
P2 plasmid (b copies)
Intracellular plasmid mixture
Memregulon promoter
Cognate inducer i
pSal Sal
pTet aTc
pBet Cho
pBAD Ara
pLux OC6
pVan Van
pTtg Nar
pCin OHC14
pTac IPTG
Memregulon i
Cm
P1
Kan
P2
Cm or Kan increase a or b
Promoter inducible by i
Ribozyme
RBS
Terminator
Weight (W) = a / (a + b)
B
Red fluorescence (AU)
pSal
pTet
pBet
pBAD
pLux
pVan
pTtg
pCin
pTac
Induced P1-only cells
2 * induced W=0.5 memregulon cells
Not detected (< 10)
Non-induced P1-only cells
2 * non-induced W=0.5 memregulon cells
C
Weight measurement in agar plates
Replica plating
Inducer/s
F-Weight
Red & green fluorescence
Growth
Parental plate
Weight
DNA quantification (qPCR/sequencing)
D
Stability of weights
Weight
Cold storage
Time (h)
pSal
pTet
pBet
pBAD
pLux
pVan
pTtg
pCin
pTac
measurement
E
Memregulon antibiotic selection in agar plates
Antibiotic + inducer/s
Parental plate
New parental plate
New parental plate

Fig. 1. Memregulons stably adapt their gene expression by varying plasmid copy-number ratios, when their promoter is activated by its cognate inducer. (**A**) Design of a *memregulon*, a stable heterozygotic plasmid system of two quasi-identical co-transformed plasmids P1 and P2, controlled by the same inducible promoter (brown). Plasmids are designed such that, in the presence of ampicillin maintenance antibiotic, the number of DNA copies of plasmids P1 and P2 (*a* and *b* respectively) remains constant in a bacterial population, maintaining the P1 plasmid fraction (weight) stable. pMB1* is a mutated medium-copy number variant of pMB1 (pMB1-med see supplementary methods). We engineered a library of 9 memregulons by using orthogonal inducible promoters (right), shown with their cognate inducers (Sal, Sodium salicylate; aTc, Anhydrotetracycline HCl; Cho, Choline chloride; Ara, L-Arabinose; OC6, 3OC6-AHL; Van, Vanillic acid; Nar, Naringenin; OHC14, 3OHC14:1-AHL; IPTG, Isopropyl-beta-D-thiogalactoside). The memregulons are designed so the use of chloramphenicol or kanamycin resistance gene (CmR or KanR), in the presence of the cognate chemical inducer and appropriate selection antibiotic, implements a positive feedback on the number of copies of the P1 or P2 plasmid. For instance, kan selection in presence of Sal will decrease the number of P1 copies of a pSal memregulon, but not of other memregulons. The plasmid maps in Fig. S18, the genetic elements in Tables S10-16. (**B**) Red fluorescence of cells containing only P1 plasmids, versus their respective memregulon systems with 0.5 weight (W), after full induction (or no induction) with their cognate inducers. AU: the memregulon's fluorescence is multiplied by 2 for better comparison with the P1-only cells. A double asterisk denotes a statistically significant ($p<0.01$) identity in fluorescence value. (**C**) The weight of memregulon cocultures grown on agar plates is measured by either fluorescence characterization (F-weight) or DNA quantification (weight). We use a replica plating procedure to copy the cultures in the parental plate to a copy plate, where they are incubated at 37 °C. For fluorescence characterization (top) we add the appropriate inducers. One inducer for a single F-weight calculation or several inducers for the sum of the corresponding memregulon's F-weights. The cells are then imaged for fluorescence expression with a blue light transilluminator, and red and green fluorescence values are obtained through image analysis software to compute the F-weight (see supplementary methods). While the copy plate is discarded after measurement, the parental plate is incubated at 37 °C for its regrowth. No inducers are added for DNA quantification (bottom). F-weights correlate well with the weights obtained by DNA sequencing (see supplementary text). Error bars indicate SD from n = 3 biological population replicates obtained on 3 different days. Data shown for a library of 9 memregulons.
(**D**) F-weights stay stable despite subsequent replica plating procedures (vertical bars). Moreover, fluorescence weights remain stable when the memregulon strains are stored for 2 weeks in 4 °C (cold storage blue band). (**E**) Memregulon weights can be changed by selection in the presence of Kanamycin/Chloramphenicol + inducers during exponential growth on agar plates. The concentrations of antibiotics are included in Table S6.

Figure 2

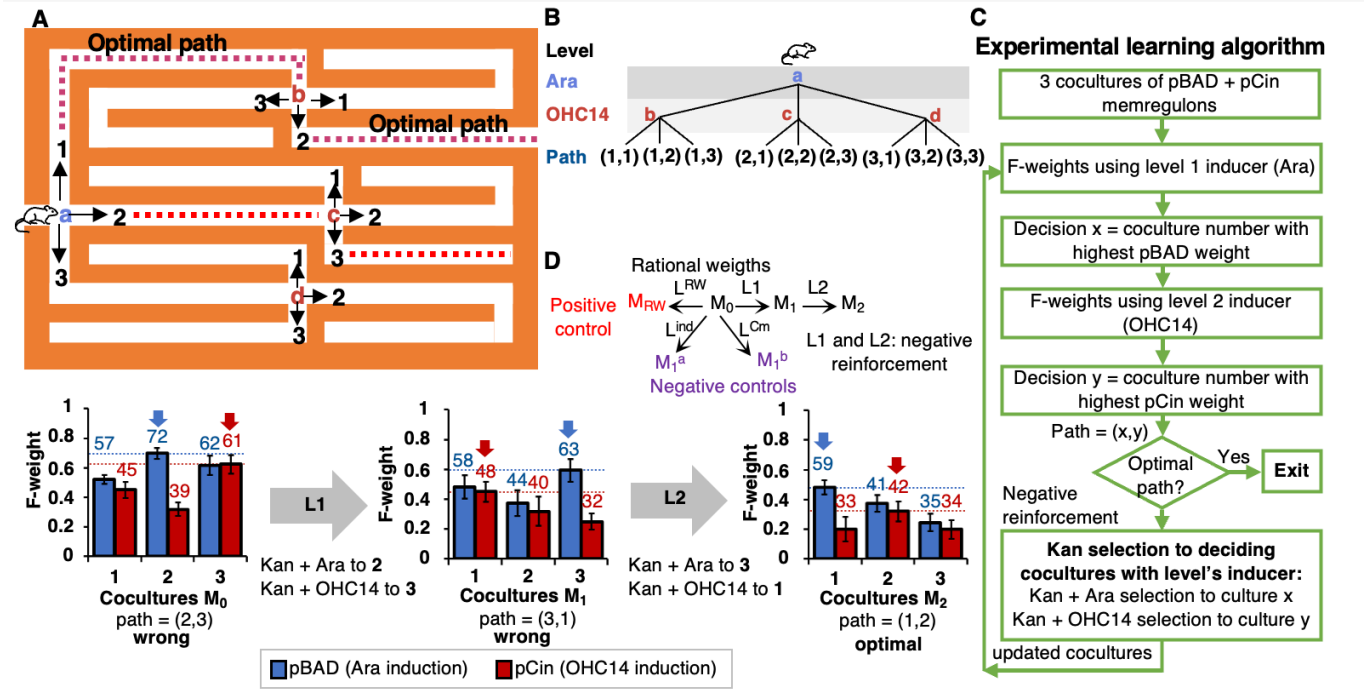
A
Optimal path
Optimal path
B
Level
Ara
OHC14
Path
(1,1) (1,2) (1,3) (2,1) (2,2) (2,3) (3,1) (3,2) (3,3)
C
Experimental learning algorithm
3 cocultures of pBAD + pCin memregulons
F-weights using level 1 inducer (Ara)
Decision x = coculture number with highest pBAD weight
F-weights using level 2 inducer (OHC14)
Decision y = coculture number with highest pCin weight
Path = (x,y)
Optimal path?
Yes
Exit
Negative reinforcement
Kan selection to deciding cocultures with level's inducer:
Kan + Ara selection to culture x
Kan + OHC14 selection to culture y
updated cocultures
D
Rational weigths
Positive control
L1 and L2: negative reinforcement
Negative controls
F-weight
Cocultures M0
path = (2,3)
wrong
L1
Kan + Ara to 2
Kan + OHC14 to 3
Cocultures M1
path = (3,1)
wrong
L2
Kan + Ara to 3
Kan + OHC14 to 1
Cocultures M2
path = (1,2)
optimal
pBAD (Ara induction)
pCin (OHC14 induction)

Fig. 2 Cocultures of two strains with different memregulons learn the decision tree of a maze by a reinforcement learning wet-algorithm. (**A**) We challenge the bacteria with a problem equivalent to finding the single optimal path in a maze of 4 crossings, with 3 diversions each, where no backtracking is allowed. The paths are described as (x, y), where x is the diversion chosen at the first crossing encountered (a) and y is the diversion chosen at the second crossing (b, c or d). (**B**) Decision tree of the maze showing the levels of the crossings, depending on the traversal order. (**C**) Flowchart with the experimental algorithm to find the optimal decisions at each crossing. We start with 3 cocultures, each composed of pBAD and pCin memregulon strains, at equal volumetric ratios but different weights. The cocultures are interrogated at each crossing by using the inducer assigned to their level, Ara or OHC14. x and y correspond to the coculture number with the highest F-weight when inducing with Ara and OHC14 respectively. The algorithm stops when the optimal path (1,2) is found, otherwise (if $x \neq 1$ or $y \neq 2$) we apply a kanamycin selection with Ara to the culture number x and with OHC14 to the coculture number y. This corresponds to a negative reinforcement operation that decreases the memregulon weights and therefore changing the followed path. The detailed experimental protocol is included in section 4 of the supplementary text. (**D**) Top: Diagram of the new coculture states (M_1 and M_2) being created after we applied successive rounds of negative reinforcement with kanamycin (L1 and L2) to the initial coculture M_0. We performed negative controls consisting of a kanamycin selection with swapped inducers (L^{ind}) and a chloramphenicol selection (L^{Cm}) to the cocultures M_0, which created M_1^a and M_1^b respectively (Fig. S6). As a positive control, we used L^{RW} to manually adjust the weights of M_0 to create cocultures M_{RW} which follow the optimal path (1,2) (Fig. S6). Bottom: F-weights measurement of the cocultures obtained by inducing with Ara and OHC14, giving the pBAD and pCin memregulon weights respectively. Left: the highest F-weights of cocultures M_0 gives the (2,3) path, which are wrong x and y values. We therefore punish the cocultures 2 and 3 with kanamycin + Ara and kanamycin + OHC14, respectively, to create the cocultures M_1. Middle: the highest F-weights of M_1 give the (3,1) path, with again incorrect x and y values. We punish the cocultures 3 and 1 with kanamycin + Ara and kanamycin + OHC14, respectively, to create the cocultures M_2. Right: the highest F-weights of M_2 give the (1,2) path, solving the maze.

Figure 3

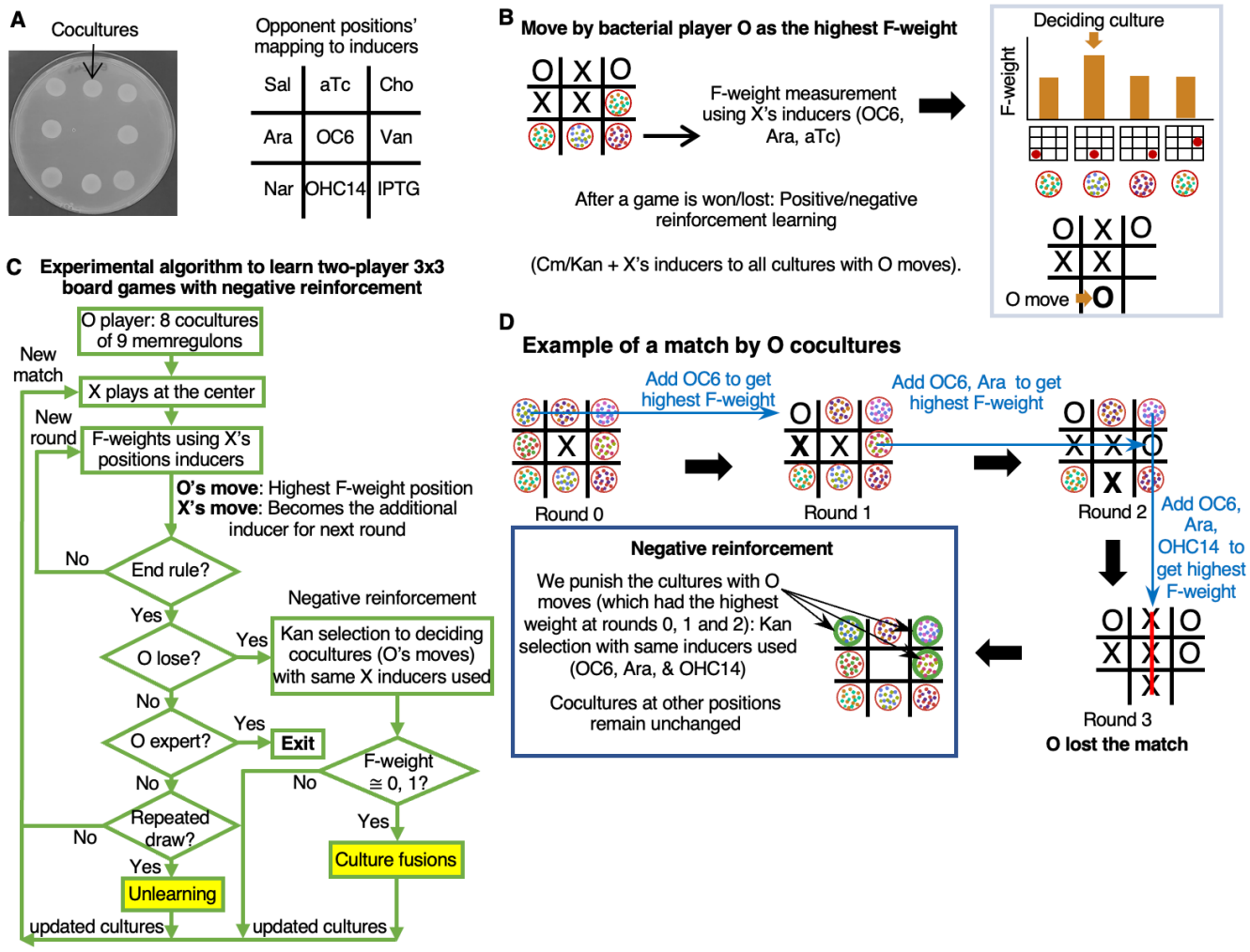
A
Cocultures
Opponent positions' mapping to inducers
Sal
aTc
Cho
Ara
OC6
Van
Nar
OHC14
IPTG
B
Move by bacterial player O as the highest F-weight
F-weight measurement using X's inducers (OC6, Ara, aTc)
After a game is won/lost: Positive/negative reinforcement learning
(Cm/Kan + X's inducers to all cultures with O moves).
Deciding culture
F-weight
O move
C
Experimental algorithm to learn two-player 3x3 board games with negative reinforcement
O player: 8 cocultures of 9 memregulons
New match
X plays at the center
New round
F-weights using X's positions inducers
O's move: Highest F-weight position
X's move: Becomes the additional inducer for next round
End rule?
No
Yes
O lose?
Yes
Negative reinforcement
Kan selection to deciding cocultures (O's moves) with same X inducers used
No
O expert?
Yes
Exit
F-weight ≅ 0, 1?
No
Yes
Culture fusions
No
Repeated draw?
No
Yes
Unlearning
updated cultures
updated cultures
D
Example of a match by O cocultures
Add OC6 to get highest F-weight
Add OC6, Ara to get highest F-weight
Round 0
Round 1
Round 2
Add OC6, Ara, OHC14 to get highest F-weight
Negative reinforcement
We punish the cultures with O moves (which had the highest weight at rounds 0, 1 and 2): Kan selection with same inducers used (OC6, Ara, & OHC14)
Cocultures at other positions remain unchanged
Round 3
O lost the match

Fig. 3. A general wet-algorithm, implementing positive and negative reinforcement learning, allowing memregulon cocultures to learn two-player 3x3 board games. (**A**) Left: Memregulon coculture arrangement in a plate for a bacterial O player. We assign cocultures at each board position except the center. Right: Mapping to inducers of the opponent positions. (**B**) A move is determined by the position of the coculture having the highest measured F-weight (Fig. 1C), when incubating the cocultures with the inducers associated to the X's moves. After the move, the parental plates are interrogated again with the new X's move. This is repeated for several rounds until the game ends. If the player O wins/loses, we apply a chloramphenicol/kanamycin selection, adding the inducers of X's positions only to the O cultures involved in the played match. (**C**) General experimental algorithm for training cocultures to learn how to play two-player 3x3 board games. The steps highlighted in yellow are not needed for tic-tac-toe. The detailed experimental protocol is included in section 6 of the supplementary text. (**D**) Example of a tic-tac-toe match showing the role played by the inducers in communicating the X's positions to the O cocultures and the selective punishment of the cocultures deciding a move in the match. X plays first at the center position and we induce all the 8 cocultures with the inducer mapped to the X's position (OC6), to compute the highest F-weight. Player O moves at the position corresponding to the coculture with the highest F-weight, here assumed to be at the top-left corner. In the following rounds, the cocultures at unoccupied positions are successively induced with all the inducers corresponding to X's positions, until the game finishes with X winning. Afterwards, we apply a negative reinforcement operation to only the cocultures at the positions that O played. This is done by a kanamycin selection in presence of the inducers of X's moves at all rounds before winning (OC6, Ara, OHC14).

Figure 4

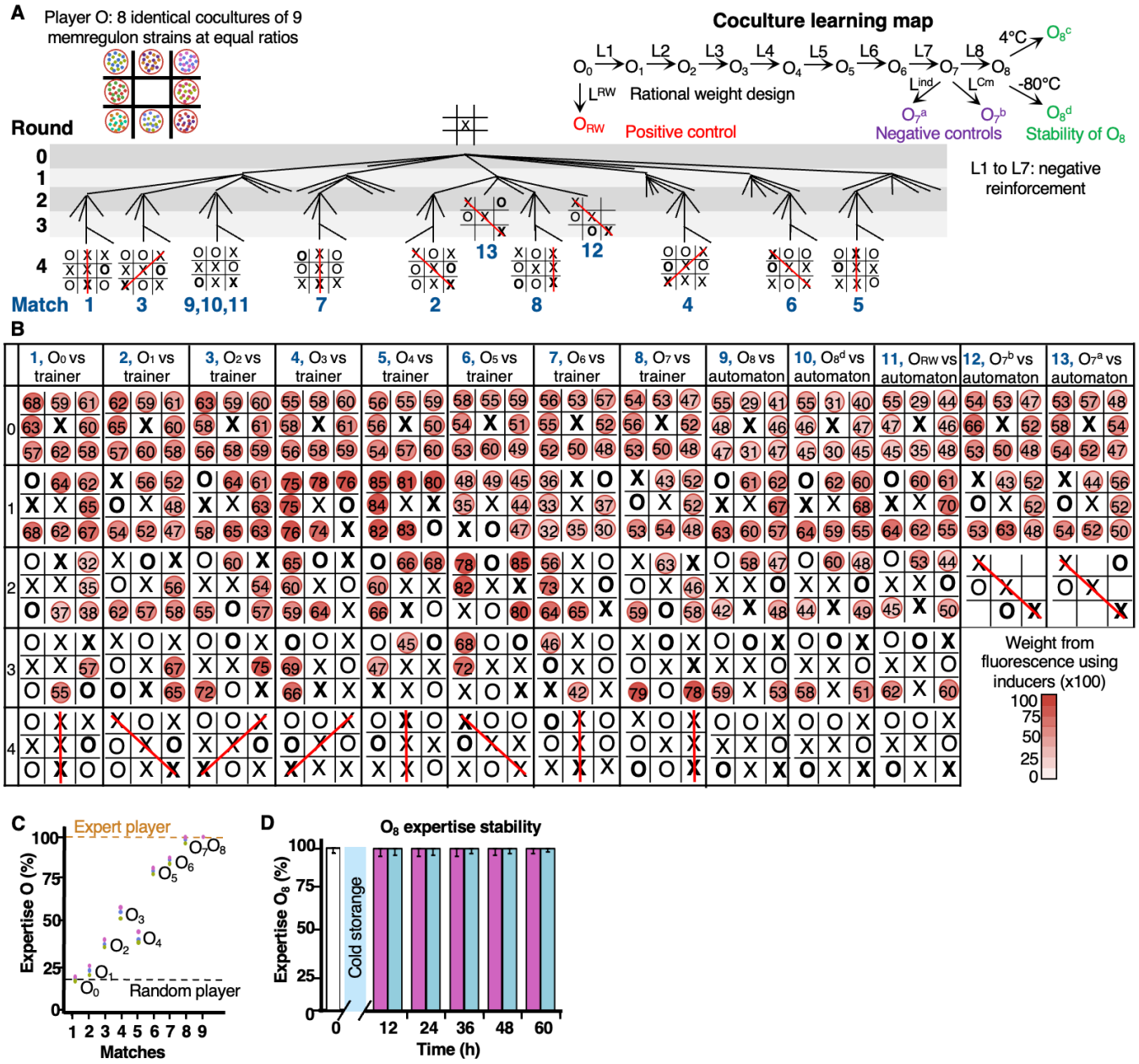
A
Player O: 8 identical cocultures of 9 memregulon strains at equal ratios
Coculture learning map
L1 L2 L3 L4 L5 L6 L7 L8
4°C
-80°C
Rational weight design
Positive control
Negative controls
Stability of O8
L1 to L7: negative reinforcement
Round
Match
B
Weight from fluorescence using inducers (x100)
C
Expert player
Random player
Expertise O (%)
Matches
D
O8 expertise stability
Cold storange
Expertise O8 (%)
Time (h)

Fig. 4. Uniform memregulon cocultures with random expertise learn to master the tic-tac-toe game. Cocultures (player O) play a tournament against a trainer (player X) designed (see supplementary text) to always try to win. (**A**) Top left: cocultures are initially setup (O_0) at each of the 8 positions by mixing (at equal volumetric ratios) memregulon strains for each of the 9 inducible promoters, and in three biological replicates. Top right: The cocultures lose at every match, and a reinforcement learning with kanamycin (L1 to L7) is applied to them, creating in succession the cocultures O_1 to O_8, until O_8 reaches 100% expertise. We use as negative controls a kanamycin selection with kanamycin and swapped inducers (L^{ind}) and one with swapped antibiotic (L^{Cm}) to the cocultures O_7, to create O_7^a and O_7^b respectively. As a positive control, we created through L^{RW} the cocultures O_{RW} by manually adjusting a coculture of 9 memregulons to have the weights of an expert player (see weight computation in the supplementary text). Bottom: Decision tree where we only show the final board game configuration for the matches experimentally played (numbered in blue). (**B**) Detail of the F-weights of the matches played (match number in blue), shown inside red-colored circles (multiplied by 100). We challenged the obtained cocultures (O_8, O_8^d, O_{RW}, O_7^a and O_7^b) to play against an expert player automaton, verifying that their matches ended in draws except for the cocultures from the negative controls. (**C**) We computed a coculture's expertise (defined as percentage of wins and draws after playing any possible match) by a computer simulation using the measured F-weights. We use a different color for each of the 3 biological replicates**.** Dashed lines indicate the random and expert players. (**D**) The mastery (100% expertise) of player O is stable in time, even after cold storage of the plates for 4 days. Error bars indicate SD from n = 3 biological population replicates, obtained on 3 different days.

References

1. A. E. Friedland *et al.*, Synthetic Gene Networks That Count. *Science* **324**, 1199-1202 (2009).
2. C. T. Fernando *et al.*, Molecular circuits for associative learning in single-celled organisms. *Journal of The Royal Society Interface* **6**, 463-469 (2009).
3. L. Yang *et al.*, Permanent genetic memory with >1 byte capacity. *Nature methods* **11**, 1261-1266 (2014).
4. A. Didovyk *et al.*, Distributed Classifier Based on Genetically Engineered Bacterial Cell Cultures. *ACS Synthetic Biology* **4**, 72-82 (2015).
5. J. Macia, B. Vidiella, R. V. Sole, Synthetic associative learning in engineered multicellular consortia. *Journal of The Royal Society Interface* **14**, 20170158 (2017).
6. P. Mohammadi, N. Beerenwinkel, Y. Benenson, Automated Design of Synthetic Cell Classifier Circuits Using a Two-Step Optimization Strategy. *Cell Syst* **4**, 207-218.e214 (2017).
7. L. B. Andrews, A. A. K. Nielsen, C. A. Voigt, Cellular checkpoint control using programmable sequential logic. *Science* **361**, eaap8987 (2018).
8. R. Zhu, J. M. del Rio-Salgado, J. Garcia-Ojalvo, M. B. Elowitz, Synthetic multistability in mammalian cells. *Science* **375**, eabg9765 (2022).
9. R. Pei, E. Matamoros, M. Liu, D. Stefanovic, M. N. Stojanovic, Training a molecular automaton to play a game. *Nature nanotechnology* **5**, 773-777 (2010).
10. L. Qian, E. Winfree, J. Bruck, Neural network computation with DNA strand displacement cascades. *Nature* **475**, 368-372 (2011).
11. P. Banda, C. Teuscher, D. Stefanovic, Training an asymmetric signal perceptron through reinforcement in an artificial chemistry. *Journal of The Royal Society Interface* **11**, 20131100 (2014).
12. X. Lin *et al.*, All-optical machine learning using diffractive deep neural networks. *Science* **361**, 1004-1008 (2018).
13. A. Pandi *et al.*, Metabolic perceptrons for neural computing in biological systems. *Nature Communications* **10**, 3880 (2019).
14. X. Li *et al.*, Synthetic neural-like computing in microbial consortia for pattern recognition. *Nature Communications* **12**, 3139 (2021).
15. K. Sarkar, D. Bonnerjee, R. Srivastava, S. Bagh, A single layer artificial neural network type architecture with molecular engineered bacteria for reversible and irreversible computing. *Chemical Science* **12**, 15821-15832 (2021).
16. M. Isalan *et al.*, Evolvability and hierarchy in rewired bacterial gene networks. *Nature* **452**, 840-845 (2008).
17. J. Carrera, S. F. Elena, A. Jaramillo, Computational design of genomic transcriptional networks with adaptation to varying environments. *Proceedings of the National Academy of Sciences of the United States of America* **109**, 15277-15282 (2012).
18. K. Sarkar, D. Bonnerjee, R. Srivastava, S. Bagh, A single layer artificial neural network type architecture with molecular engineered bacteria for complex conventional and reversible computing. *BioRxiv*, (2021).
19. J. Schrittwieser *et al.*, Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature* **588**, 604-609 (2020).

20. W. Tang, D. R. Liu, Rewritable multi-event analog recording in bacterial and mammalian cells. *Science* **360**, eaap8992 (2018).
21. A. J. Meyer, T. H. Segall-Shapiro, E. Glassey, J. Zhang, C. A. Voigt, Escherichia coli "Marionette" strains with 12 highly optimized small-molecule sensors. *Nature chemical biology* **15**, 196-204 (2019).
22. L. Chua, Memristor-The missing circuit element. *IEEE Transactions on Circuit Theory* **18**, 507-519 (1971).
23. D. Krotov, J. J. Hopfield, Unsupervised learning by competing hidden units. *Proceedings of the National Academy of Sciences* **116**, 7723-7731 (2019).
24. S. Regot *et al.*, Distributed biological computation with multicellular engineered networks. *Nature* **469**, 207-211 (2011).
25. A. Tamsir, J. J. Tabor, C. A. Voigt, Robust multicellular computing using genetically encoded NOR gates and chemical 'wires'. *Nature* **469**, 212-215 (2011).
26. M. N. Stojanovic, D. Stefanovic, A deoxyribozyme-based molecular automaton. *Nature biotechnology* **21**, 1069-1074 (2003).
27. W. Maass, On the computational power of winner-take-all. *Neural Comput* **12**, 2519-2535 (2000).
28. A. E. Elo, *The rating of chessplayers, past and present*. (BT Batsford Limited, 1978).
29. Y. Cao, J. Yang, in *2015 IEEE Symposium on Security and Privacy (SP)*. (IEEE, 2015), pp. 463-480.
30. A. Engel, C. Van den Broeck, *Statistical Mechanics of Learning*. (Cambridge University Press, 2012), pp. 343.
31. D. O. Hebb, *The organization of behavior: a neuropsychological theory*. (Science editions, 1949).
32. C. Koch, *Biophysics of computation: information processing in single neurons*. (Oxford university press, 2004).
33. N. Frémaux, W. Gerstner, Neuromodulated Spike-Timing-Dependent Plasticity, and Theory of Three-Factor Learning Rules. *Frontiers in Neural Circuits* **9**, (2016).
34. C. A. Voigt, S. Kauffman, Z. G. Wang, Rational evolutionary design: the theory of in vitro protein evolution. *Adv. Protein Chem.* **55**, 79-160 (2000).
35. W. Kong, D. R. Meldgin, J. J. Collins, T. Lu, Designing microbial consortia with defined social interactions. *Nature chemical biology* **14**, 821-829 (2018).
36. C. C. Guet, M. B. Elowitz, W. Hsing, S. Leibler, Combinatorial synthesis of genetic networks. *Science* **296**, 1466-1470 (2002).
37. K. M. Cherry, L. Qian, Scaling up molecular pattern recognition with DNA-based winner-take-all neural networks. *Nature* **559**, 370-376 (2018).
38. M. Ibba, P. Kast, H. Hennecke, Substrate specificity is determined by amino acid binding pocket size in Escherichia coli phenylalanyl-tRNA synthetase. *Biochemistry* **33**, 7107-7112 (1994).
39. L. S. Qi *et al.*, Repurposing CRISPR as an RNA-guided platform for sequence-specific control of gene expression. *Cell* **152**, 1173-1183 (2013).
40. Y. Li *et al.*, In vitro evolution of enhanced RNA replicons for immunotherapy. *Scientific Reports* **9**, 6932 (2019).
41. A. Jaramillo, Engineered stable ecosystems. *Nat Microbiol* **2**, 17119 (2017).
42. R. Mathis, M. Ackermann, Response of single bacterial cells to stress gives rise to complex history dependence at the population level. *Proceedings of the National Academy of Sciences of the United States of America* **113**, 4224-4229 (2016).

43. J. Rodriguez-Beltran *et al.*, Multicopy plasmids allow bacteria to escape from fitness trade-offs during evolutionary innovation. *Nat Ecol Evol* **2**, 873-881 (2018).
44. C. Hildenbrand, T. Stock, C. Lange, M. Rother, J. Soppa, Genome Copy Numbers and Gene Conversion in Methanogenic Archaea. *Journal of Bacteriology* **193**, 734-743 (2011).
45. J. Aryaman, I. G. Johnston, N. S. Jones, Mitochondrial Heterogeneity. *Frontiers in Genetics* **9**, 718 (2019).