Homework 2 – ECS 289, Winter 2021

Required problems

1. **Composition.** Recall that the following CRN with input species X_1, X_2 and output species Y stably computes $y = x_1 - x_2$ whenever $x_1 \geq x_2$:

$$\begin{aligned} X_1 &\to Y \\ X_2 + Y &\to \varnothing \end{aligned}$$

And the following CRN with input species Y, X_3 and output species Z stably computes $z = \min(y, x_3)$

$$Y + X_3 \to Z$$

   (a) Prove that the CRN obtained by simply combining the above 3 reactions, with input species X_1, X_2, X_3 and output species Z, does *not* stably compute $z = \min(x_1 - x_2, x_3)$ (assuming $x_1 \geq x_2$). Going back to the definition of stable computation, the proof should demonstrate a concrete counterexample to the claim "*for all* $\mathbf{x}$ *reachable from the initial configuration* $\mathbf{i}$*, there is a correct, stable* $\mathbf{o}$ *reachable from* $\mathbf{x}$."

   Also, intuitively describe the *reason* that this CRN fails to stably compute that function.

   (b) Design a CRN that stably computes $z = \min(x_1 - x_2, x_3)$ (assuming $x_1 \geq x_2$).

2. **Electing more than one leader.** Consider the following CRN, which stably computes $f(x) = 1$:

$$\begin{aligned} X &\to Y \\ 2Y &\to Y \end{aligned}$$

   (a) Design a leaderless CRN that stably computes $f(x) = 2$, in which every reaction has at most two reactants and two products.

   (b) Describe how to generalize the previous CRN to produce, for each $k \in \mathbb{N}$, a leaderless CRN that stably computes the function $f(x) = k$, in which every reaction has at most two reactants and two products. Try to do this with as few reactions as possible. ($O(k)$ reactions works, but $O(\log k)$ is possible.)

3. **Stable majority with tie-detection.** We showed a CRN in lecture that stably computes the predicate $\phi(a, b) = \text{Yes} \iff a \geq b$. We called it "majority", even though it considers a the majority even in the case of a tie.

   Design a 3-output CRN that stably computes majority and properly detects ties. By "3-output", we mean that each species "votes" for one of A (for $a > b$), B (for $a < b$), or T

(for $a = b$), and the CRN should stably decide which of these three is the case by reaching a stable configuration where all present species agree on the correct vote.

Analyze its time complexity in three cases: 1) $a = b$, 2) $a = b + 1$, and 3) $a = 2b$.

4. **Chemical caucusing.** Imagine we have two candidates, A and B, being voted on by $n \in \mathbb{Z}^+$ caucus-goers. If A and B supporters encounter each other, one becomes undecided. Undecided voters are swayed by decided voters:

$$\begin{aligned} A + B &\rightarrow A + U \\ B + A &\rightarrow B + U \\ A + U &\rightarrow 2A \\ B + U &\rightarrow 2B \end{aligned}$$

I lived in Iowa for 12 years, and this is a mostly accurate summary of how the caucus works.

(a) Show that for any initial configuration $\mathbf{i}$ with $\|\mathbf{i}\| = n$ and $\mathbf{i}(A) + \mathbf{i}(B) > 0$, for all $\mathbf{c}$ such that $\mathbf{i} \Longrightarrow \mathbf{c}$, there is a configuration $\mathbf{o}$ such that $\mathbf{c} \Longrightarrow \mathbf{o}$ and either $\mathbf{o}(A) = n$ or $\mathbf{o}(B) = n$. That is, the CRN is guaranteed eventually to reach a stable consensus.

(b) Deriving how the expected time scales as a function of n is remarkably difficult. (Or, there is a short, elegant proof that has not yet been discovered.)

Run simulations for different initial configurations and compute the time to stabilize to a consensus. How does the time scale with different initial population sizes, and how is it affected by different initial distributions of A, B, and U?

To simulate, you have two alternatives:

i. Write the simulation code yourself; this is probably the simplest option. I'd recommend using Python with a Jupyter notebook and use Matplotlib to generate the plots. (Or print something easy to export to MS Excel or Google Sheets to generate plots.) Since this is a population protocol, the discrete-time population protocol model is the simplest to simulate. Pick a random pair of molecules to interact, and count it as an interaction even if they have a "null" interaction (e.g., two B's). Divide the total number of interactions by n to measure time.

ii. Use an existing simulator for CRNs/population protocols, for example:

Peregrine: `https://peregrine.model.in.tum.de/` Population protocol simulator. I haven't used it, so I can't vouch for how easy it is to use or to generate.

VisualDSD: `https://classicdsd.azurewebsites.net/` Originally it was designed for DNA strand displacement, but it also has a CRN simulator if you click on the "CRN" tab at the top left. It has the ability to generate plots of counts over time. The syntax is not straightforward, so I'd recommend the above options over this one. Load one of the examples to see the syntax.

Actually, this CRN *is* one of the examples, called AM - Stochastic (*approximate majority*). But to measure time correctly, you have to set the volume to the total count n using the "scale" parameter via `directive stochastic {scale = 100; steps=1000;};` for $n = 100$ to run for time 1000, and species initial counts as concentrations, e.g., `A=0.6` instead of `A=60`. see `https://www.microsoft.com/en-us/research/uploads/prod/2009/02/Visual_DSD_Manual.pdf`.

Optional problems

You don't have to do these, but I think they are interesting to think about.

1. **Combining function and predicate computation.** Design a CRC that stably computes the function
$$f(x_1, x_2) = \begin{cases} x_1, & \text{if } x_1 \geq x_2; \\ 0, & \text{otherwise.} \end{cases}$$

2. **Composition of other functions.** Recall that the following CRN with input species X_1, X_2 and output species Y stably computes $y = \max(x_1, x_2)$:
$$\begin{aligned} X_1 &\rightarrow Y + W_1 \\ X_2 &\rightarrow Y + W_2 \\ W_1 + W_2 &\rightarrow K \\ K + Y &\rightarrow \varnothing \end{aligned}$$
And the following CRN with input species Y, X_3 and output species Z stably computes $z = \min(y, x_3)$
$$Y + X_3 \rightarrow Z$$
   (a) Show that the CRN obtained by simply combining the above 5 reactions, with input species X_1, X_2, X_3 and output species Z, does *not* stably compute $z = \min(\max(x_1, x_2), x_3)$. Intuitively describe the *reason* that this CRN fails to stably compute that function.

   (b) Design a CRN that stably computes $z = \min(\max(x_1, x_2), x_3)$.

3. **Fast stable computation.** Design a CRN that stably computes $f(n_1, n_2) = n_2$ if $n_1 > 0$ and $f(n_1, n_2) = 0$ otherwise. It should reach a stable configuration in expected time $O(\log n)$, where $n = n_1 + n_2$, and the volume is n.

4. **Clear majority wins.** In problem 4, show that if the initial configuration is $\mathbf{i} = \{0.51n\, B, 0.49n\, H\}$, then for sufficiently large n, with probability at least 99%, the CRN reaches the configuration $\{nB\}$. That is, the initial majority probably wins.

5. **Large-state protocols.** It is known that leaderless population protocols as defined in lecture are unable to compute most predicates and functions in sublinear time. Three examples are majority (the predicate "$a > b$?"), parity (the predicate "is a odd?"), and leader election (the function $f(x) = 1$). However, the proof relies on the assumption that the number of states is $O(1)$, independent of the population size n.

   Consider allowing the number of states and allowed transitions to change with the population size n. In this case, pick one of the above problems (or all of them if you are feeling ambitious), and give a polylogarithmic-time population protocol for stably computing it.

   Hint: It is helpful to imagine that each agent has the value $\lceil \log n \rceil$ hard-coded into it, so that it is possible to do reactions such as $X_i + C \rightarrow X_{i+1} + C$ if $i < \lceil \log n \rceil$ and $X_i + C \rightarrow X' + C$ if $i = \lceil \log n \rceil$.

6. **Consuming output.** Some CRNs, such as $X \to 2Y$ and $2X \to Y$, only produce the output species. Others, such as $X_1 \to Y$, $X_2 + Y \to \varnothing$ to compute subtraction, do consume the output species.

   (a) Show that if $f : \mathbb{N}^k \to \mathbb{N}$ is not a monotone function, then any CRN stably computing f must consume the output species Y, i.e., must have a reaction where Y's stoichiometric coefficient as a reactant exceeds its stoichiometric coefficient as a product (e.g., $X+Y \to A$ or $3Y \to 2Y$).

   (b) Show that any leaderless CRN stably computing $f(x_1, x_2) = \max(x_1, x_2)$ must consume the output species Y.

   (c) Show that any CRN, even with a leader, stably computing $f(x_1, x_2) = \max(x_1, x_2)$ must consume the output species Y. **Hint:** use Dickson's Lemma.