Output-Oblivious Stochastic Chemical Reaction Networks

Ben Chugg[1,2] Anne Condon[2] Hooman Hashemi[2]

[1]University of Oxford
[2]University of British Columbia

Oxford, June 2019

Acknowledgements

Independent work by Severson, Haley, and Doty (2019).

Full paper in Proceedings of OPODIS 2018.

Chemical Reaction Networks (CRNs)

Remember high school (A-levels?) chemistry?

Chemical Reaction Networks (CRNs)

Remember high school (A-levels?) chemistry?
CRN:

- Finite set of *species* Λ. E.g., $\Lambda = \{A, B, C, D\}$.
- Finite set of *reactions* $\mathcal{R}$, E.g., $A + B \to C$.

Chemical Reaction Networks (CRNs)

Remember high school (A-levels?) chemistry?
CRN:

- Finite set of *species* Λ. E.g., $\Lambda = \{A, B, C, D\}$.
- Finite set of *reactions* $\mathcal{R}$, E.g., $A + B \to C$.

That's it! (Almost ...)

Why YOU should care

- Closely related to distributed computing models such as Population Protocols. Applications in Signal Processing, Graphical Models and modelling cellular processes.

http://www.quickmeme.com/meme/3qmru3

Why YOU should care

- Closely related to distributed computing models such as Population Protocols. Applications in Signal Processing, Graphical Models and modelling cellular processes.
- "Programming Language" for molecular programming and DNA computing. Implementation via Strand Displacement Systems.

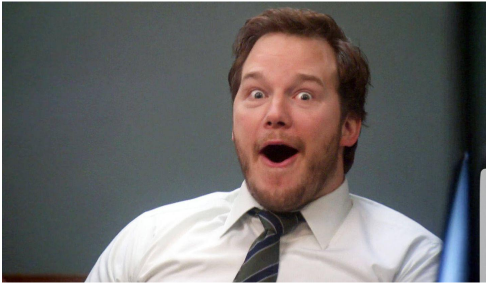
Why YOU should care

- Closely related to distributed computing models such as Population Protocols. Applications in Signal Processing, Graphical Models and modelling cellular processes.
- “Programming Language” for molecular programming and DNA computing. Implementation via Strand Displacement Systems.
- Mathematically fun ...

https://steemit.com/@funmeme

CRN Function Computation

We can view CRNs as computing functions!



https://www.memesmonkey.com/topic/chris+pratt

CRN Function Computation

We can view CRNs as computing functions!

$$X \to 2Y$$
$$Y + L \to \emptyset$$

CRN Function Computation

We can view CRNs as computing functions!

$$X \to 2Y$$
$$Y + L \to \emptyset$$

Initially n copies of X, 1 copy of L (the "leader").

Computes the function $f(n) = 2n - 1$.

CRN Function Computation

We can view CRNs as computing functions!

$$X \to 2Y$$
$$Y + L \to \emptyset$$

Initially n copies of X, 1 copy of L (the "leader").

Computes the function $f(n) = 2n - 1$.
Function computation is *stable*: will always produce correct answer, no matter the order of computation.

CRN Function Computation

Precisely the *semilinear* functions are stably computable. (Chen, Doty, and Soloveichik 2014).

CRN Function Computation

Precisely the *semilinear* functions are stably computable. (Chen, Doty, and Soloveichik 2014).

Essentially, piecewise affine functions. E.g.,

$$f(n) = \begin{cases} 3, & n \leq 5 \\ 3n - 2, & n \text{ even}, \ n \geq 5 \\ 2n + 6, & \textit{otherwise}. \end{cases}$$

Composable CRNs?

If CRNs are implemented in real systems, want them to be *composable*.

Composable CRNs?

If CRNs are implemented in real systems, want them to be *composable*.

Order in which reactions occur is stochastic! Could have a second CRN $\mathcal{C}_2$ consume output of $\mathcal{C}_1$ before $\mathcal{C}_1$ was done.

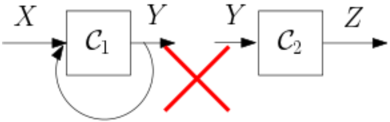
https://me.me/t/houston-we-have-a-problem

Composable CRNs?

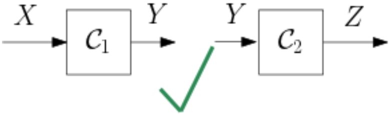
If CRNs are implemented in real systems, want them to be *composable*.

Order in which reactions occur is stochastic! Could have a second CRN $\mathcal{C}_2$ consume output of $\mathcal{C}_1$ before $\mathcal{C}_1$ was done.

Leads to incorrect results :(.

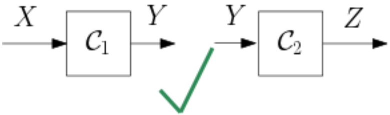


We can avoid this travesty if the CRNs are *output-oblivious*: Outputs are never reactants.



Output-Oblivious CRNs

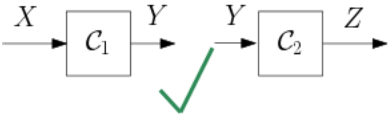
We can avoid this travesty if the CRNs are *output-oblivious*: Outputs are never reactants.



Main research question: Which functions admit output-oblivious CRNs?

Output-Oblivious CRNs

We can avoid this travesty if the CRNs are *output-oblivious*: Outputs are never reactants.



Main research question: Which functions admit output-oblivious CRNs?

Call these output-oblivious functions.

Output-Oblivious Functions

Focus on functions $f : \mathbb{N}^2 \to \mathbb{N}$.

Output-Oblivious Functions

Focus on functions $f : \mathbb{N}^2 \to \mathbb{N}$.

Clearly f cannot be decreasing.

Output-Oblivious Functions

Focus on functions $f : \mathbb{N}^2 \to \mathbb{N}$.

Clearly f cannot be decreasing.

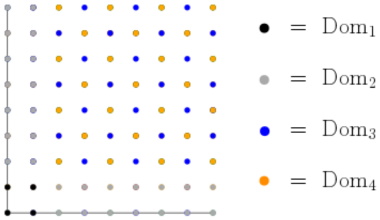
One class of output-oblivious functions: *Grid affine functions*.

These are piecewise affine functions defined on different “grids” (which can be zero-, one-, or two-dimensional).

Grid-Affine functions

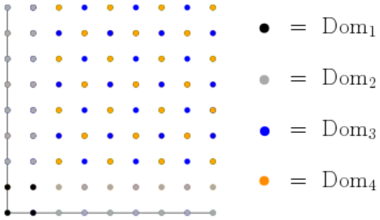
Example:

$$f(\boldsymbol{n}) = \begin{cases} 2, & \boldsymbol{n} \in \mathrm{Dom}_1 \cup \mathrm{Dom}_2 \\ 2n_1 + 3n_2 + 1, & \boldsymbol{n} \in \mathrm{Dom}_3 \\ 2n_1 + 3n_2, & \boldsymbol{n} \in \mathrm{Dom}_4. \end{cases}$$



Example:

$$f(\boldsymbol{n}) = \begin{cases} 2, & \boldsymbol{n} \in \mathrm{Dom}_1 \cup \mathrm{Dom}_2 \\ 2n_1 + 3n_2 + 1, & \boldsymbol{n} \in \mathrm{Dom}_3 \\ 2n_1 + 3n_2, & \boldsymbol{n} \in \mathrm{Dom}_4. \end{cases}$$



Intuition: You can "track" on which grid the input lies, since grids are (roughly) a constant distance from one another.

Any other takers?

What about min?

Any other takers?

What about min?

$\min(n_1, n_2)$ is output-oblivious:

$$X_1 + X_2 \to Y.$$

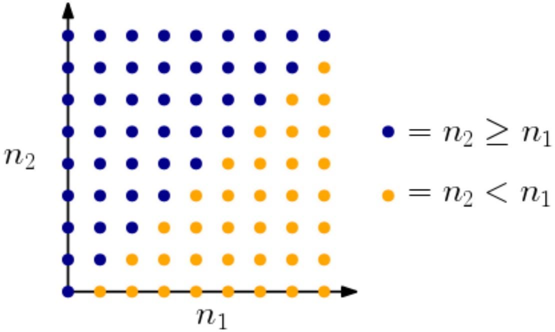
Any other takers?

What about min?

$\min(n_1, n_2)$ is output-oblivious:

$$X_1 + X_2 \to Y.$$

But not grid-affine!



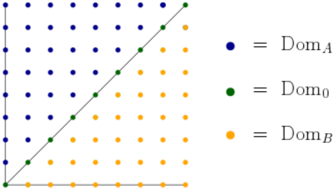
“Min-like” functions

Enter *fissure functions*: These are “min-like”; they agree with the min function everywhere except on “fissure lines.”

"Min-like" functions

Enter *fissure functions*: These are "min-like"; they agree with the min function everywhere except on "fissure lines."

$$f(\boldsymbol{n}) = \begin{cases} 3n_1 + 2n_2 + 1, & \boldsymbol{n} \in \mathrm{Dom}_A, \\ 5n_1, & \boldsymbol{n} \in \mathrm{Dom}_0, \\ 2n_1 + 3n_2 + 2, & \boldsymbol{n} \in \mathrm{Dom}_B. \end{cases}$$



Here f agrees with $\min\{2n_1 + 3n_2 + 2, 3n_1 + 2n_2 + 1\}$, except on the "fissure line" Dom_0.

The main result!

If f_1 and f_2 are output-oblivious, then $\min\{f_1 + f_2\}$ is output oblivious:

$\mathcal{C}_1$ computes f_1 and produces Y_1.
$\mathcal{C}_2$ computes f_2 and produces Y_2.

$$\min(f_1, f_2) : \quad Y_1 + Y_2 \to Z.$$

The main result!

If f_1 and f_2 are output-oblivious, then $\min\{f_1 + f_2\}$ is output oblivious:

$\mathcal{C}_1$ computes f_1 and produces Y_1.
$\mathcal{C}_2$ computes f_2 and produces Y_2.

$$\min(f_1, f_2): \quad Y_1 + Y_2 \to Z.$$

Therefore, since it's illegal to conclude a theory talk without stating a confusing theorem.

Theorem

An increasing, semilinear function $f : \mathbb{N}^2 \to \mathbb{N}$ is stably computable by an output-oblivious CRN if and only if it is grid-affine or the minimum of finitely many fissure functions.