The Synergy of Finite State Machines

Yuval Emek

Technion — Israel Institute of Technology

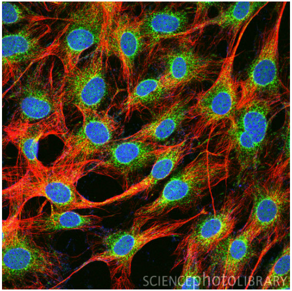
The 22nd International Conference on Principles of Distributed Systems
18-Dec-2018
Hong Kong

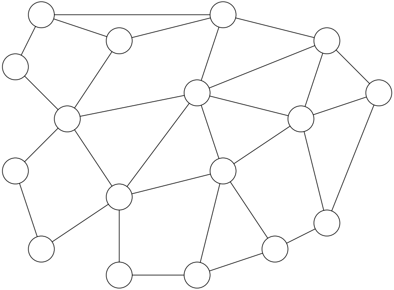
Joint work with Yehuda Afek and Noa Kolikant

Synopsis

- Motivation: what can be computed by a biological cellular network?
  - Network rather than individual cells
  - "What" before "how fast"
- Requirement 1: abstract model for a biological cellular network
  - Variant of stone age model [Emek & Wattenhofer 2013]
- Requirement 2: network as a computational device
  - Receive input, perform computation, return output

Stone age model: DistComp in networks of weak devices

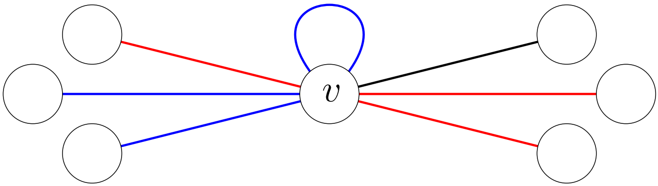




- Undirected graph $G = (V, E)$
- Nodes communicate by exchanging messages
  - Weak communication scheme
- Nodes process incoming data by performing local computation
  - Weak computational model
- This work: bounded node degrees
  - Maximum degree $\Delta = O(1)$
  - Most "physically deployed" networks

The Communication Scheme

- Synchronous scheduler
  - Paper: asynchronous scheduler
- In every round, each node transmits a message from fixed Σ
- For each $m \in \Sigma$, node $v \in V$ distinguishes between
  - 0 nodes in $N(v)$ transmit m
  - ≥ 1 node in $N(v)$ transmits m
- Set-broadcast [Hella et al. 2015], beeping [Cornejo & Kuhn 2010]
- Graph may include self-loops: $v \in N(v)$
  - No sender collision detection
  - Going beyond [EW13]



The Local Computation Scheme

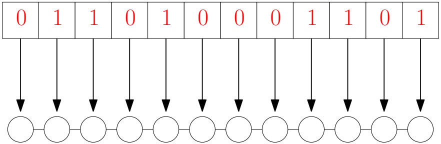
- Computational power of single cell not fully understood
- Model choice: nodes run randomized finite state machine
  - Supported by [Benenson et al. 2001]
- Fixed state space Q
  - $O(1)$ bits of memory
- Node transition function $\phi : Q \times \{0, 1\}^{\Sigma} \rightarrow Q \times \Sigma$
  - Domain: current state and incoming messages
  - Range: next state and transmitted message
  - Allow randomness
- Crux of SA model:
  - All nodes run same randomized finite state machine
  - $|Q|$ and $|\Sigma|$ are constants

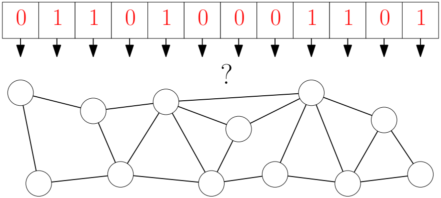
The mission: $\mathrm{RSPACE}(n)$

- [EW13]:
  - Simulate n-node SA network by $\mathrm{RSPACE}(n)$ machine
    - $=$ randomized Turing machine with tape of size n
  - Simulate $\mathrm{RSPACE}(n)$ by SA network on n-node path
- What can be computed by n-node SA network of arbitrary topology?
- Computational power inherently $\leq \mathrm{RSPACE}(n)$
  - $O(n)$ space for all nodes combined
- Main question:
  Can n-node SA network of arbitrary topology simulate $\mathrm{RSPACE}(n)$?

Providing the input

- Input $\mathcal{I}$ of $\mathrm{RSPACE}(n)$ machine $\mathcal{M}$ is bitstring of size n
  - Placed in $\mathcal{M}$'s tape at beginning of computation
- How is $\mathcal{I}$ deployed in n-node SA network?
- Trivial with path topology
- But we deal with arbitrary topology...
  - Nodes are not canonically ordered





Sequential SA machines

- Solution: feed input bitstring $\mathcal{I}$ to SA network, bit-by-bit
- Sequential SA machine = SA algorithm allowing external user to
  1. Pick any node $v^{\mathtt{io}} \in V$ and send to it `I/O-prepare` message
  2. Wait until $v^{\mathtt{io}}$ transmits `I/O-ready` message
  3. Feed $\mathcal{I}$ to $v^{\mathtt{io}}$, bit-by-bit
  4. Wait until computational process terminates
  5. Get output back from $v^{\mathtt{io}}$, bit-by-bit
- T^p = time between sending `I/O-prepare` and receiving `I/O-ready`
- T^s = time between feeding input bits and getting back output bits
- Challenges of SSAM designer:
  - Distribute $\mathcal{I}$ over $\Omega(n)$ nodes during step 3
    - $O(1)$ space per node
  - Simulate execution of $\mathcal{M}$ on $\mathcal{I}$ during step 4
  - Small preparation time T^p and simulation time T^s

Positive result

Theorem

Any problem that can be solved w.h.p. by $\mathrm{RSPACE}(n)$ *machine in time* T *can be solved w.h.p. by SSAM on any n-node bounded degree graph with* $T^p = O(\mathrm{diameter})$ *and* $T^s = O(T)$.

- Main algorithmic contribution:
  SA algorithm running during preparation time T^p
- Algorithm constructs 2-hop coloring in G
- Algorithm constructs skeleton node sequence $\langle S(i) \rangle_{i=0}^{2n-1}$ so that
  - Every node appears in S exactly twice
  - $S(0) = S(2n-1) = v^{\mathtt{io}}$
  - $S(i)$ can route a message to $S(i \pm 1 \bmod 2n)$ in $O(1)$ time
    - $S(i)$ cannot store i
- Skeleton S is employed to
  - Distribute $\mathcal{I}$ during step 3
  - Simulate $\mathrm{RSPACE}(n)$ machine's tape during step 4
    - Similar to simulating tape over path topology [EW13]

Negative result

- Two simplifying assumptions:
  - Node degrees bounded by universal constant Δ
  - Algorithm provided with designated "leader" ($v^{\mathtt{io}}$)
- Natural question: are the assumptions necessary?
- Trivial: 1st assumption cannot be avoided for 2-hop coloring
  - Nodes with $O(1)$ space cannot store "large" colors
- Nor can 2nd assumption...

Theorem

Without a designated node (v^{io}), any randomized SA algorithm that constructs a 2-hop coloring must fail w.p. that tends to 1 as $n \to \infty$.

- Holds even for 1-hop coloring

Source of difficulty: 2-hop coloring

- 2-hop coloring known to be very useful
  - Enables local IDs and unicast
  - Key to routing messages along skeleton S
- Challenging to obtain under SA model (even with bounded degrees)
  - More complicated due to self-loops
- Node v cannot verify (w.p. 1) that $N(v)$ free of color conflicts
- Solution:
  - Color nodes concurrently with growing tree $\mathcal{T}$ rooted at $v^{\mathtt{io}}$
    - $\mathrm{depth}(\mathcal{T}) = \Theta(\mathrm{diameter})$
  - Run randomized tests (repeatedly) to detect color conflicts
  - Use $\mathcal{T}$ to reset (and restart) if color conflicts detected
    - Tree structure ensures resets terminate safely
- (Eventually: skeleton S = DFS traversal of $\mathcal{T}$)

Source of difficulty: correctness w.h.p.

- Algorithm should succeed w.h.p.
  - Algorithm fails w.p. $\leq n^{-c}$ for arbitrarily large constant c
- SA model: individual nodes don't have any notion of n
- How can we obtain w.h.p. guarantees under SA model?
- Solution:
  Ensure that each $v \in V$ runs $\geq \mathrm{depth}(\mathcal{T})$ color conflict tests
  - When $\mathcal{T}$ is constructed, root initiates B&E
  - v keeps running color conflict tests as long as B&E isn't over
- Why is it good enough?
  - $\Delta = O(1)$ implies $\mathrm{diameter} \geq \Omega(\log n)$
  - $\implies \mathrm{depth}(\mathcal{T}) = \Theta(\mathrm{diameter}) \geq \Omega(\log n)$
  - Each randomized test detects color conflicts in $N(v)$ w.p. $\Omega(1)$
  - $\implies v$ detects color conflicts w.h.p.
  - Union bound: all nodes detect color conflicts w.h.p.
- Paper: if 2-hop coloring succeeds, then whole algorithm succeeds

Wrapping up

- What can computed by a biological cellular network?
  - Stone age model
  - Bounded degrees
  - Self-loops
- Introducing sequential stone age machines
  - Input provided sequentially, bit-by-bit, rather than all at once
- During preparation time:
  - Construct 2-hop coloring
  - Construct skeleton S
    - Enables simulation of $(2n)$-cell tape
- Paper: $q \in Q$ and $m \in \Sigma$ encoded using $O(\log \Delta)$ bits
- Can we deal with multiple $v^{\mathtt{io}}$ nodes?
  - DISC18: SA leader selection among bounded #candidates

Open questions

- Computational power of SA algorithms in unbounded degree graphs
  - Cannot construct 2-hop coloring, but perhaps it can be avoided?
  - Conjecture: computational power $< \mathrm{RSPACE}(n)$
- Beyond SA model:
  - DistComp in networks with low-memory nodes
  - More accurate abstractions for biological cellular networks
    - Can we deal with noise?

谢谢