Population Protocols and Generalizations

Othon Michail
Paul Spirakis
Ioannis Chatzigiannakis

Research Academic Computer Technology Institute
(RACTI)

May 2009

Outline I

Formal Definition of Population Protocols

[Angluin, Aspnes, Diamadi, Fischer, and Peralta, PODC '04]

Population V of $|V| = n$ agents. A PP $\mathcal{A}$ consists of

- finite **input and output alphabets** X and Y,
- finite set of **states** Q,
- **input function** $I : X \to Q$,
- **output function** $O : Q \to Y$,
- **transition function** $\delta : Q \times Q \to Q \times Q$.

$\delta(p, q) = (p', q')$ or simply $(p, q) \mapsto (p', q')$ is called a **transition**.

Significant Properties

- **Uniformity:** Protocol descriptions are independent of the population size n.
- **Anonymity:** There is no room in the state of an agent to store a unique identifier.

Fairness Assumption

- A **communication graph** $G = (V, E)$ describes the permissible interactions.
- First graph considered: **all-pairs** (complete directed communication graph)
- Model passive movement by an **adversary scheduler** that picks members of E.
- Make the scheduler "*computation-friendly*" by a fairness assumption
  - Do not allow avoidance of a possible step forever.

Computation

- $C : V \to Q$, **population configuration** specifying the state of each agent
  - $C \xrightarrow{e} C'$, C can go to C' in one step via encounter $e \in E$.
  - $C \to C'$, C can go to C' in one step, i.e there exists $e \in E$ s.t. $C \xrightarrow{e} C'$.
  - $C \xrightarrow{*} C'$, C' is reachable from C (there exists sequence $C = C_0, C_1, \ldots, C_k = C'$ s.t. $C_i \to C_{i+1}$ for all $0 \leq i < k$).

Execution: Finite or infinite sequence $C_0, C_1, C_2, \ldots$, s.t. $C_i \to C_{i+1}$ for all i.
Fairness Formally: For all C, C' s.t. $C \to C'$, if C occurs infinitely often in the execution the same holds for C'.
Computation: Infinite fair execution.

Why use in Sensor Nets

- Each agent is severely limited.
  - Constant storage capacity (independent of n).
- The designer cannot control the interactions between agents.
- Models societies of networked mobile tiny artefacts.
  - Limited sensing, signal processing and communication capabilities.
  - Limited energy.
- Pervasive, adaptive and scalable systems.

Stable Computation

- **Population protocols do not halt**.
- No fixed time to view the output of the population.
- Instead we talk about **stability**.
- Stability ensures that the computation reaches a point after which no agent can change its output.

Stable Computation

- $\mathcal{X}$: the set of all input assignments.
- $\mathcal{Y}$: the set of all output assignments.
- If $x \in \mathcal{X}$ then $I(x)$ is the **input configuration** corresponding to x.
- Each configuration C has a corresponding **output assignment** $O(C) = y_C \in \mathcal{Y}$.

Stable Computation

- C is **output-stable** if $O(C') = O(C)$ for all C' reachable from C (no agent changes its output).
- A computation that contains an output-stable configuration C **stabilizes to output** y_C.

Stable Computation

- A PP $\mathcal{A}$ **stably computes a function** $F_{\mathcal{A}} : \mathcal{X} \to \mathcal{Y}$ iff for every $x \in \mathcal{X}$ and every computation that starts from $I(x)$ (initial configuration) the computation reaches an output-stable configuration C, where $O(C) = F_{\mathcal{A}}(x)$.
- If $F_{\mathcal{A}} : \mathcal{X} \to \{0, 1\}$ and $Y = \{0, 1\}$, then we call $F_{\mathcal{A}}$ a **stably computable predicate**.
  - **predicate output convention**: we require that all agents eventually agree on the correct output value.

Problem Description

"*Find if at least 5 sensors have detected elevated temperature.*"

- Each agent senses the temperature of a distinct bird after a global start signal.
- If detected elevated temperature input 1, else 0 (i.e. $X = \{0, 1\}$).
- We want every agent to eventually output
  - 1, if at least 5 birds were found sick,
  - 0, otherwise.

A Protocol

- $X = Y = \{0, 1\}$, $Q = \{q_0, q_1, \ldots, q_5\}$,
- $I(0) = q_0$ and $I(1) = q_1$,
- $O(q_i) = 0$, for $0 \leq i \leq 4$, and $O(q_5) = 1$,
- δ:

$$(q_i, q_j) \rightarrow (q_{i+j}, q_0), \text{ if } i + j < 5$$
$$\rightarrow (q_5, q_5), \text{ otherwise.}$$

Why it Works...

- Due to fairness, all agents with non-zero state index will eventually interact with each other.
- In each such interaction one of them keeps the sum.
- All indices are eventually aggregated in one agent's state index j (assuming that no faults can happen).
- If $j < 5$, then q_5 cannot occur, thus no agent ever outputs 1.
- Otherwise, state q_5 appears and floods the population (2nd rule), i.e. eventually every agent outputs 1.

Semilinear Predicates

An example: The input multiset $\{a, a, a, b, b\}$ over the input alphabet $X = \{a, b, c\}$ can be represented by the vector $(3, 2, 0)$ ($N_a = 3$, $N_b = 2$ and $N_c = 0$)

- A **semilinear set** S is a subset of $\mathbb{N}^d$.
- Finite union of linear sets of the form

$$\{\vec{b} + k_1\vec{a_1} + k_2\vec{a_2} + \cdots + k_m\vec{a_m} \mid k_i \in \mathbb{N}\}$$

  $\vec{b}$: base vector, $\vec{a_i}$: basis vectors and k_i: non-negative integer coefficients.
- A predicate on input assignments is **semilinear** if its support (input assignments mapped to 1) is a semilinear set.

Presburger Arithmetic
[Presburger 1929]

- Arithmetic on natural numbers with addition but not multiplication.
- Allows **quantifier elimination**.
- Formulas involving addition, $<$, mod-k congruence relation $\equiv_k$ for each constant k and usual logical connectives $\vee$, $\wedge$ and $\neg$.

Semilinear sets are those that can be defined in Presburger arithmetic [Ginsburg and Spanier, 1966].

Exact Characterization

- Any semilinear predicate is stably computable in the basic population protocol model [Angluin et al. 2004].
- Any predicate stably computable in the basic population protocol model is semilinear [Angluin et al. 2006]

Theorem

A predicate is computable in the basic population protocol model if and only if it is semilinear.

Some Examples...

Stably Computable (semilinear)

- "The number of a's is greater than 5" (i.e. $N_a > 5$).
- $(N_a = N_b) \vee (\neg(N_b > N_c))$.
- $((17N_1 \geq 3N_0) \wedge (4N_1 \leq N_0))$ ("The number of 1's is between 15% and 20% of the total population").
- "An odd number of a's occur".

Non-stably computable (non-semilinear)

- "The number of c's is the product of the number of a's and the number of b's" (i.e. $N_c = N_a \cdot N_b$).
- $N_1 = 2^d$.
- $(N_b < N_a^2) \wedge (N_c \geq N_b^3)$.

Formal Definition of Mediated Population Protocols

[I. Chatzigiannakis, O. Michail, and P. G. Spirakis, ICALP '09]

Population V of $|V| = n$ agents forming a communication graph $G = (V, E)$. A MPP $\mathcal{A}$ consists of

- finite **input and output alphabets** X and Y,
- finite set of **agent states** Q, **agent input function** $I : X \longrightarrow Q$, **agent output function** $O : Q \longrightarrow Y$,
- finite set of **edge states** S, **edge input function** $\iota : X \longrightarrow S$, **edge output function** $\omega : S \longrightarrow Y$,
- **output instruction** r,
- (**totally ordered cost set** K, **cost function** $c : E \longrightarrow K$) optional, and
- **transition function** $\delta : Q \times Q \times K \times S \longrightarrow Q \times Q \times K \times S$.

Mediated Population Protocols

Transition function $\delta : Q \times Q \times S \rightarrow Q \times Q \times S$

- Assume that costs are not defined.
- When agents u_1, u_2 in states a, b, respectively, interact through (u_1, u_2) in state s then $(a, b, s) \rightarrow (a', b', s')$ is applied and
  - a goes to a',
  - b goes to b', and
  - s goes to s'.

New Assumptions about Edges...

- We assume that each edge is equipped with a buffer of $\mathcal{O}(1)$ storage capacity (independent of the population).
  - **Each pair of communicating agents shares a memory of constant size**.
- During interaction (u, v) the corresponding agents read the memory contents and update it according to δ.
- In most cases we try to find protocols that only need 1 or 2 bits of edge storage.

New Assumptions about Edges...

- If costs are defined, during interaction (u, v) the agents u, v also read $c((u, v))$ but do not modify it.
- K is the codomain of c (totally ordered).
- Usually, we assume that $K \subset \mathbb{Z}^+$ and $c_{max} = \max_{w \in K} \{w\} = \mathcal{O}(1)$. (**all costs are storable**).

In this case any $u \in V$ is capable of storing at most kc_{max} cumulative costs, where $k = \mathcal{O}(1)$ and we say that the cost function is **useful**.

- A **network configuration** is a mapping $C : V \cup E \rightarrow Q \cup S$ specifying the agent state of each agent in the population and the edge state of each edge in the communication graph.

r-stability

- **Problem**: "*Given an undirected communication graph* $G = (V, E)$ *and a useful cost function* $c : E \to K$ *on the set of edges, design a protocol that will find the minimum cost edges of* E".
- **Example of instruction** r: "*Get each* $e \in E$ *for which* $\omega(s_e) = 1$ *(where* s_e *is the state of* e*)*".

r-stable network configuration C: for every C', (i) If a subgraph needs to be found, C fixes a subgraph that doesn't change in any C' reachable from C (ii) If a function has to be computed by the agents, then an r-stable configuration drops down to an agent output-stable configuration.

- **Permits protocols that search for and, eventually, find certain subgraphs of** G**.**

Stable Computation

Definition

A protocol $\mathcal{A}$ **stably solves** a problem Π, if for every instance I of Π and every computation of $\mathcal{A}$ on I, the network reaches an r-stable configuration C that gives the correct solution for I if interpreted according to the output instruction r. If Π is a function f to be computed we say instead that $\mathcal{A}$ **stably computes** f.

Definition

In the special case where Π is an optimization problem (like *minimum cost edges*), a protocol that stably solves Π will be called an **optimizing population protocol** for problem Π.

Maximal Matching

Problem

*(**Maximal matching**) Given an undirected communication graph $G = (V, E)$, find a **maximal matching**, i.e., a set $E' \subseteq E$ such that no two members of E' share a common end point in V and, moreover, there is no $e \in E - E'$ such that e shares no common end point with every member of E'.*

A protocol

MaximalMatching

- $X = \{0\}$,
- $Y = \{0, 1\}$,
- $Q = \{q_0, q_1\}$,
- $I(0) = q_0$,
- $O(q_0) = 0,\ O(q_1) = 1$,
- $S = \{0, 1\}$,
- $\iota(0) = 0$,
- $\omega(0) = 0,\ \omega(1) = 1$,
- r: "*Get each* $e \in E$ *for which* $\omega(s_e) = 1$ *(where* s_e *is the state of* e*)*",
- δ:

$$(q_0, q_0, 0) \rightarrow (q_1, q_1, 1)$$

Proof of correctness

Theorem

Protocol *MaximalMatching* **stably solves the maximal matching problem**.

Proof.

M: set of edges in state 1. Two interactions happening in parallel cannot concern adjacent edges (natural assumption about scheduler). $e = (u, v)$ gets in M iff both endpoints are in q_0. Then u, v go to q_1 to indicate adjacency with an edge of M. So any edge conflicting with M cannot get in M and M is always a matching. Any edge $e' = (u', v')$ not conflicting with M will eventually get in M because of fairness and because $q(u') = q(v') = q_0$ and $s(e') = 0$, so M eventually will become maximal. □

MPP is stronger than PP

- Obviously, PP model is a special case of MPP model.
  - Ignore edge functions, states and costs to get PP.
- The edge buffers enable each pair of agents to remember a **pairwise history of up to $\mathcal{O}(1)$ interactions**.
- Predicates concerning the **cardinality of the edges** (satisfying some property) are now possible.
- Construction of subgraphs like long paths and cycles becomes also possible
  - Chosen edges can be identified by their different output value.

For Example...

Assume the all-pairs directed graph $G = (V, E)$.

- $N_c = N_a \cdot N_b$.
- Rephrase it: "**Is N_c equal to the number of edges leading from agents with input a to agents with input b?**"
- Complete graph, thus the number of these links equals the product to be computed.
- This predicate is **not semilinear**, since Presburger arithmetic does not allow multiplication of variables.

$N_c = N_a \cdot N_b$ protocol

VarProduct

- $X = \{a, b, c, 0\}$,
- $Y = \{0, 1\}$,
- $Q = \{a, \dot{a}, b, c, \bar{c}, 0\}$,
- $I(x) = x$, for all $x \in X$,
- $O(a) = O(b) = O(\bar{c}) = O(0) = 1$, and $O(c) = O(\dot{a}) = 0$,
- $S = \{0, 1\}$,
- $\iota(x) = 0$, for all $x \in X$,
- r: "*If there is at least one agent with output 0, reject, else accept.*",
- δ:

$$(a, b, 0) \to (\dot{a}, b, 1)$$
$$(c, \dot{a}, 0) \to (\bar{c}, a, 0)$$
$$(\dot{a}, c, 0) \to (a, \bar{c}, 0)$$

Proof Sketch

- The number of links leading from agents in state a to agents in state b equals $N_a \cdot N_b$.
- For each a the protocol tries to erase b c's.
- Each a is able to remember the b's that has already counted (for every such b a c has been erased) by marking the corresponding links.
- If the c's are less than the product then at least one $\dot{a}$ remains and if the c's are more at least one c remains. In both cases at least one agent that outputs 0 remains.
- If $N_c = N_a \cdot N_b$ then every agent eventually outputs 1.

Discussion

- A MPP **strongly stably computes** a predicate if in every computation all agents eventually agree on the correct output value.
- This is not the case for *VarProduct* (sometimes only one agent eventually gives output 0 and the answer is "reject").
- But it is easy to see that that *VarProduct* has **stabilizing states**:
  - In every computation all agents eventually stop changing their state (stronger than stabilizing outputs).
- Moreover, instruction r defines a **semilinear predicate on multisets of *VarProduct*'s states** (we can write it formally as $(N_c > 0) \vee (N_{\dot{a}} > 0)$).
- We exploit these properties to prove that with a slight modification *VarProduct* strongly stably computes $N_c = N_a \cdot N_b$.

Composition Theorem

Theorem

Any MPP $\mathcal{A}$, that stably computes a predicate p with stabilizing states in some family of directed and connected communication graphs $\mathcal{G}$, containing an instruction r that defines a semilinear predicate t on multisets of $\mathcal{A}$'s agent states, can be composed with a provably existing MPP $\mathcal{B}$, that strongly stably computes t with stabilizing inputs in $\mathcal{G}$, to give a new MPP $\mathcal{C}$ satisfying the following properties:

- *$\mathcal{C}$ is formed by the composition of $\mathcal{A}$ and $\mathcal{B}$,*
- *its input is $\mathcal{A}$'s input,*
- *its output is $\mathcal{B}$'s output, and*
- *$\mathcal{C}$ strongly stably computes p in $\mathcal{G}$.*

Non-uniform Upper Bounds

Let *DMP* be the class of predicates stably computable by the MPP model in any family of directed communication graphs.

Theorem

All predicates in DMP are also in NSPACE(m).

Proof

Any network configuration can be represented explicitly by storing a state per node and a state per edge. This takes $\mathcal{O}(m)$ space since G is always connected. Any stably computable predicate corresponds to a unique stably computable language (its support). Let L be a stably computable language (and $\mathcal{A}$ the MPP that computes it). We present a NTM $M_{\mathcal{A}}$ that decides L in space $\mathcal{O}(m)$. To accept input x, $M_{\mathcal{A}}$ must verify two conditions:

1. *That there exists a configuration C reachable from $I(x)$ satisfying r.*
2. *There is no C' reachable from C in which r is violated.*

Non-uniform Upper Bounds

The first condition is verified by guessing C_{i+1} to replace C_i. Obviously, a C satisfying r can be found by using at most $\mathcal{O}(m)$ space in any branch of the $M_{\mathcal{A}}$'s computation. The second condition is the complement of the similar reachability problem: "There is some C' reachable from C in which r is violated" and NSPACE is closed under complement for all space functions $\geq \log n$ [Immerman 1988]. □

Summary

- **Population Protocol model** was the **first step** in this widely unexplored field of societies of tiny networked artefacts.

Our research:

- **Mediated Population Protocol model**: A natural extension of the PP model gives birth to a **promising new area of research**.
- Many new directions: **finding subgraphs**, **deciding graph properties**, **optimization**...
- Moreover the MPP model is computationally stronger than the Population Protocol model.
- **Verification** is the key for applying such protocols in real, critical systems.

FRONTS

- This work has been partially supported by the ICT Programme of the European Union under contract number ICT-2008-215270 (FRONTS).
- FRONTS is a joint effort of eleven academic and research institutes in foundational algorithmic research in Europe.
- A major goal is to establish the **foundations of adaptive networked societies of tiny artefacts**.

Thank You!