Challenges in Population Protocols

Janna Burman

Colloquium on Distributed Algorithms
Collège de France, 2019
Population Protocols (PP)  
[Angluin et al. PODC’04, DC’06]

- Collection (*population*) of computational *agents*
  - of unknown size *n*
  - uniform (indistinguishable)
  - finite state, independent of *n* (constant)
  - anonymous

- Interacting
  - in asynch. and unpredictable way
  - in pairs, while exchanging and updating their states according to a *transition function*

*Example of a protocol: compute a global property* (predicate/function) *eventually* on the input values of the agents
  - E.g., whether 10% of the population have an elevated input value?  

$$\text{Transition} \quad (p,q) \rightarrow (p',q')$$

$$\text{predicate } P(x,y,z,w,...)$$
Motivating scenarios

- **Passively mobile sensor networks**
  - ZebraNet [ASPLOS’02] (wildlife tracking)
  - EMMA [WCMC’07] (pollution monitoring)

- **Social networks**
  - propagation of:
    - trust [Diamadi, Fischer WU.J.Nat.Sci.01]
    - epidemics [Bailey, 75] [Herbert et al, SIAM’00]

- **Chemical Reaction Networks**
  - dynamics of well mixed solutions
    - [Gellespie 77], [SoloveichikCookWinfreeBruck 08], [Doty SODA’2014]

- **Game Theory**
  - repetitive games of n-participants
    - [Bournez, Chalopin, Cohen, Koegler, Rabie OPODIS’11]
Interaction graph and fairness

**Interaction Graph**
- nodes = agents
- edge \((u,v)\) = possible interaction
- weakly connected
- Frequently a complete graph

**Fairness**
- Weak: each pair of agents interacts infinitely often
- Global: infinitely often reachable configuration is reached infinitely often
- Probabilistic: each pair interacts uniformly at random

Probabilistic Fairness \(\Rightarrow\) Global Fairness w.p.1
Main complexity measures in PP

**Space** complexity: in number of different possible *memory states* of an agent.

**Time** complexity *with probabilistic fairness*: in terms of expected *parallel interactions* (1 parallel = \( n \) consecutive interactions) *until stabilization* (to the correct output/behavior).
PP – Minimalist Model

- PP compute a predicate $P \iff P$ is semi-linear eq. 1st order formula in Presburger arithmetic [Angluin et al. DC’07]*

* holds even with $o(\log \log n)$ memory bits [Chatzigiannakis, Michail, Nikolaou, Pavlogiannis, Spirakis TCS’11]
PP – Minimalist Model

- **Termination is impossible** (only eventual stabilization)

- **Fault-tolerance is limited**:
  - O(1) crash and transient faults can be tolerated [Delporte-Gallet, Fauconnier, Guerraoui, Ruppert DCOSS’06]
  - Any number of transient faults (self-stabilization) is frequently impossible to tolerate (leader election [Cai, Izumi, Wada TCS’12], phase clock [Beouquier, Burman DCOSS’10], counting [Beauquier, Clement, Messika, Rosaz, Rozoy DISC’07], bipartition [Yasumi, Ooshita, Yamaguchi, Inoue – OPODIS’17, …])
  - Communication faults are impossible to tolerate [Luna, Flochini, Izumi, Izumi, Santoro, Viglietta TCS’19]
  - Byzantine tolerant protocols are impossible [Guerraoui & Ruppert ICALP’09]

- **Stabilization time acceleration is limited**
  - Every semi-linear predicate computable in O(n) parallel time [Angluin, Aspnes, Eisenstat DC’08], and some (e.g., majority) cannot be computed faster [Belleville, Doty, Soloveichik ICALP’2018]
  - Leader Election takes Ω(n) parallel time [Doty & Soloveichik DISC’15]
Extensions to obtain termination

- Relaxing the termination requirement
  - eventual stabilization may be sufficient
    - depending on an application
    - composing non-terminating protocols is possible [Angluin, Aspnes, Chan, Fischer, Jiang, Peralta DCOSS’15]

- Oracles
  - “heard of all” detector for solving consensus [Beauquier, Blanchard, Burman, Kutten AlgoSensors’15]
  - “state absence” detector based leader → allow terminating PP with Turing Machine power of space O(log n) [Michail & Spirakis J PDC’15]
Extensions to augment computational power

- With $\Theta(\log \log n)$ memory bits eq. $\Theta(\log^{O(1)} n)$ identifiers (homonyms)
  - the first non-semi-linear predicate can be computed
    [Chatzigiannakis, Michail, Nikolaou, Pavlogiannis, Spirakis TCS’11]
  - allows to simulate Turing Machine on space $O(\log^{O(1)} n)$
    [Bournez, Cohen, Rabie TCS’18]

- Adding unique identifiers - $\Omega(\log n)$ memory bits
  (Community Protocols or Passively mobile Machines model) $\rightarrow$
  symmetric predicates in NSPACE($n \log n$) eq. to a power of TM
  with $O(n \log n)$ space [Guerraoui & Ruppert ICALP’09],
  [Chatzigiannakis, Michail, Nikolaou, Pavlogiannis, Spirakis TCS’11]

- Adding shared memory per agent pair (Mediated Population Protocols) $\rightarrow$
  symmetric predicates in NSPACE($n^2$) eq. to TM
  with $O(n^2)$ space [Chatzigiannakis, Michail, Nikolaou, Pavlogiannis, Spirakis ICALP’09]
Extensions for speed up

- With a **given leader** constant-space PP (semi-linear predicates) **converge** exponentially faster – $O(\text{polylog } n)$ parallel time [Angluin, Aspnes, Eisenstat DC’08], [Belleville, Doty, Soloveichik ICALP’2018]

- With a **small probability of error** constant-space PP **converge** in $O(\text{polylog } n)$ parallel time [Kosowski & Uznanski]
Extensions for fault-tolerance

- Adding unique identifiers - $\Omega(\log n)$ memory bits - Community Protocols - $O(1)$ Byzantine faults can be tolerated [Guerraoui & Ruppert ICALP’09]

- With a leader and/or unbounded memory some communication faults can be tolerated [Luna, Flochini, Izumi, Izumi, Santoro, Viglietta TCS’19]

- What about any transient number of faults - self-stabilization?
**Self-stabilization** [Dijkstra’74]

**Motivation:** any number of transient failures, hard to initialize, agents that leave and join

**Self-stabilizing protocol:**
starting from an **arbitrary configuration**, reaches (barring additional faults) **correct configurations** eventually (and stays correct)

Fault-tolerance to transient faults $\rightarrow$ **Self-Stabilization**

Fault attack
Self-stabilizing PP
[Angluin, Aspnes, Fischer ACMJ ’08]

Positive results:
- coloring, orientation, spanning-tree in bounded degree graphs
- non-uniform Leader Election (LE) in rings

Negative result:
- uniform LE in complete graphs is impossible

→ No general characterization of self-stabilizing PP
Extensions for fault-tolerance

Self-stabilizing LE

- with "leader absence detector" - oracle $\Omega$?
  - uniform leader election in rings [Fischer & Jiang OPODIS’06]
  - uniform leader election in arbitrary graphs [Beauquier, Blanchard, Burman OPODIS’13, SSS’16] [Canepa & Potop-Butucaru WRAS’10]
Extensions for fault-tolerance

Self-stabilizing LE (cont.)

- With \textit{n states} and knowledge of \textit{n}  
  \cite{Cai, Izumi, Wada TCS’12}
  - \(O(n^2)\) time solution
  - impossible otherwise

- With \textit{stronger models} and less than \textit{n states}  
  \begin{itemize}
  \item mediated PP \cite{Mizoguchi, Ono, Kijima, Yamashita DC’12}
  \item k-interaction PP \cite{Xu, Yamauchi, Kijima, Yamashita SSS’13}
  \end{itemize}

- With \textit{upper bound \textbf{N on n} and relaxed self-stabilization} - loose-stabilization  
  \begin{itemize}
  \item With \(\exp(N)\) holding time: stabilization \(\Omega(Nn)\) and \(\Omega(N)\) states are necessary and sufficient \cite{Izumi SIROCCO’15}
  \item Solution stabilizing in polylog\((n)\) time but with poly\((n)\) holding time \cite{Sudo, Ooshita, Kakugawa, Masuzawa, Datta, Larmore OPODIS’18}
  \end{itemize}
Self-stabilizing LE vs. Initialized LE

- While impossible without initialization, easy with uniform initialization
  - with one bit of memory
  - one transition rule \((\text{leader, leader}) \rightarrow (\text{leader, non-leader})\)
    (when two candidate leaders meet, one drops out)

- The best SS-LE stabilizes in \(O(n^2)\) time – exponentially slower than \(\text{polylog}(n)\) time initialized LE

- Very few studies on self-stabilizing PP!
Future directions: self-stab. PP

- Study time efficiency limits (time-space trade-offs) of self-stab. LE
- Study other self-stab. PP
  (majority, counting, naming …)
- General characterization of n-state self-stab. PP
Future Population Protocols

- Adapt to new applications (e.g. more nature inspired)
  - position aware PP
  - beeping PP
  - PP implementing micro-biological circuits
    - future biological computers
    - intelligent drugs
Why Population Protocols?

- Simple and convenient model allowing formal analysis
  - Can be extended

- Model many real world phenomena
  - Many existing and future applications

- Still many open algorithmic questions
  - Related to model, problems and complexity

Thank you!