Self-Stabilizing Broadcast with O(1)-Bit Messages

Emanuele Natale†

joint work with
Lucas Boczkowski* and Amos Korman*

4th Workshop on Biological Distributed Algorithms (BDA)
July 25-29, 2016
Chicago, Illinois

*preprint at goo.gl/ETNc64
Self-Stabilizing Broadcast with $O(1)$-Bit Messages*  
(Bit Dissemination)

Emanuele Natale†

joint work with
Lucas Boczkowski* and Amos Korman*

4th Workshop on Biological Distributed Algorithms  
(BDA)  
July 25-29, 2016  
Chicago, Illinois

*preprint at goo.gl/ETNc64
Bit Dissemination Problem
Bit Dissemination Problem
Examples

Flocks of birds
[Ben-Shahar et al. ’10]
Examples

Flocks of birds
[Ben-Shahar et al. ’10]

Schools of fish
[Sumpter et al. ’08]
Examples

Flocks of birds
[Ben-Shahar et al. ’10]

Schools of fish
[Sumpter et al. ’08]

Insects colonies
[Franks et al. ’02]
Communication Model

Animal communication:
- Chaotic
- Anonymous
- Passive
- Parsimonious
**Communication Model**

Animal communication:
- Chaotic
- Anonymous
- Passive
- Parsimonious

**PULL\((h, \ell)\) model** [Demers ’88]: at each round each agent can *observe* \(h\) other agents chosen independently and uniformly at random, and *shows* \(\ell\) bits to her observers.
Communication Model

Animal communication:
✓ Chaotic
✓ Anonymous
✓ Passive
✓ Parsimonious

\textit{PULL}(h, \ell) model [Demers '88]: at each round each agent can observe \( h \) other agents chosen independently and uniformly at random, and shows \( \ell \) bits to her observers.
Communication Model

Animal communication:
- Chaotic
- Anonymous

\textbf{\textit{PULL}}(\(h, \ell\)) model
[Demers ’88]: at each round each agent can observe \(h\) other agents chosen independently and uniformly at random, and shows \(\ell\) bits to her observers.
Communication Model

Animal communication:
- Chaotic
- Anonymous
- Passive
- Parsimonious

**PULL** \((h, \ell)\) model [Demers ’88]: at each round each agent can observe \(h\) other agents chosen independently and uniformly at random, and shows \(\ell\) bits to her observers.
Sources’ bits (and other agents’ states) may change in response to *external environment*
(Probabilistic) Self-Stabilization

Sources’ bits (and other agents’ states) may change in response to *external environment*
(Probabilistic) Self-Stabilization

Sources’ bits (and other agents’ states) may change in response to *external environment*
Sources’ bits (and other agents’ states) may change in response to *external environment*.

blue vs red: \(\frac{39}{14} \approx 2.8\)
(Probabilistic) Self-Stabilization

(Probabilistic) self-stabilization:

\[ S := \{ \text{"correct configurations of the system"} \} \]

(= consensus on source’s bit)

- **Convergence.** From *any* initial configuration, the system reaches \( S \) (w.h.p.)
- **Closure.** If in \( S \), the system stays in \( S \) (w.h.p.)

*(Probabilistic) Self-stabilizing* algorithm:

guarantees *convergence* and *closure* w.r.t. \( S \) (w.h.p.)
(Self-Stab.) Bit Dissemination vs Synchronization
(Self-Stab.) Bit Dissemination vs Synchronization
(Self-Stab.) Bit Dissemination vs Synchronization
(Self-Stab.) Bit Dissemination vs Synchronization
(Self-Stab.) Bit Dissemination vs Synchronization
(Self-Stab.) Bit Dissemination vs Synchronization
(Self-Stab.) Bit Dissemination vs Synchronization
(Self-Stab.) Bit Dissemination vs Synchronization

Self-stabilizing algorithms converge from any initial configuration.
Self-Stabilizing Clock Sync. in the \textit{PULL} Model

2-Majority dynamics [Doerr et al. ’11]. Converge to consensus in $O(\log n)$ rounds with high probability.
Self-Stabilizing Clock Sync. in the \textbf{PULL} Model

2-Majority dynamics [Doerr et al. ’11]. Converge to consensus in $O(\log n)$ rounds with high probability.
Self-Stabilizing Clock Sync. in the **PULL** Model

2-Majority dynamics [Doerr et al. ’11]. Converge to consensus in $O(\log n)$ rounds with high probability.
Self-Stabilizing Clock Sync. in the **PULL** Model

2-Majority dynamics [Doerr et al. ’11]. Converge to consensus in $O(\log n)$ rounds with high probability.
Self-Stabilizing Clock Sync. in the \textbf{PULL} Model

Message Reduction Lemma

\begin{center}
\begin{tikzpicture}
\node[draw] (P) at (0,0) {\textbf{P}};
\node[draw] (EmulP) at (0,-2) {\textbf{EMUL}(P)};
\node[draw] (Public) at (0,-4) {Public};
\node[draw] (Private) at (0,-6) {Private};
\node[draw] (Public2) at (0,-8) {Public};
\node[draw] (Private2) at (0,-10) {Private};
\node[draw] (Message) at (0,-12) {\text{message reduction lemma}};
\node[draw] (bits) at (0,-14) {\text{bits}};
\node[draw] (logbits) at (0,-16) {\text{log} \ell + 1 \text{ bits}};
\node[draw] (10100101) at (0,-4) {10100101};
\node[draw] (01111) at (0,-8) {01111};
\node[draw] (10100101) at (0,-12) {10100101};
\end{tikzpicture}
\end{center}
Self-Stabilizing Clock Sync. in the **PULL** Model

Message Reduction Lemma

**EMUL(P)**

\[ \log \ell + 1 \]

\[ \gamma \]

\[ \begin{array}{cccccccc}
1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 1 & 0 & 0 & 1 & 0 & 1 & 0
\end{array} \]

**SYN-CLOCK**

\[ \begin{array}{cccc}
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{array} \]

\[ \begin{array}{ccccccccc}
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & \cdots & 1 & 0 & 1 & 0
\end{array} \]
Results

Theorem (Clock Synchronisation). SYN-CLOCK is a self-stabilizing clock synchronization protocol which synchronizes a clock modulo $T$ in $\tilde{O}(\log n \log T)$ rounds w.h.p. using 3-bit messages.

Theorem (Self-Stabilizing Bit Dissemination). There is a self-stabilizing Bit Dissemination protocol which converges in $\tilde{O}(\log n)$ rounds w.h.p. using 3-bit messages.
Self-Stab. Bit Diss. with 1 bit: a Candidate

**BFS**\((f, s)\). Agents can *boosting*, 1/0-*frozen* or 1/0-*sensitive*.

- **Boosting**: Update their opinion with majority of their bit and the 2 bits they pull. If they see only agents of color \(c\) for \(s\) rounds, they become \(c\)-sensitive.
- **c-sensitive**: Turn into \(c\)-frozen if see value \(c\).
- **c-frozen** keep value \(c\) for \(f\) rounds before becoming *boosting*. 
Self-Stab. Bit Diss. with 1 bit: a Candidate

\( \text{BFS}(f, s) \). Agents can **boosting**, 1/0-**frozen** or 1/0-**sensitive**.

- **Boosting**: Update their opinion with majority of their bit and the 2 bits they pull. If they see only agents of color \( c \) for \( s \) rounds, they become **\( c \)-sensitive**.
- **\( c \)-sensitive**: Turn into **\( c \)-frozen** if see value \( c \).
- **\( c \)-frozen** keep value \( c \) for \( f \) rounds before becoming **boosting**.
BFS($f, s$). Agents can *boosting*, 1/0-*frozen* or 1/0-*sensitive*.

- **Boosting**: Update their opinion with majority of their bit and the 2 bits they pull. If they see only agents of color $c$ for $s$ rounds, they become $c$-sensitive.
  - **c-sensitive**: Turn into $c$-frozen if see value $c$.
  - **c-frozen** keep value $c$ for $f$ rounds before becoming *boosting*. 
Self-Stab. Bit Diss. with 1 bit: a Candidate

BFS($f, s$). Agents can \textit{boosting}, 1/0-\textit{frozen} or 1/0-\textit{sensitive}.

- \textit{Boosting}: Update their opinion with majority of their bit and the 2 bits they pull. If they see only agents of color $c$ for $s$ rounds, they become $c$-\textit{sensitive}.
- $c$-\textit{sensitive}: Turn into $c$-\textit{frozen} if see value $c$.
- $c$-\textit{frozen} keep value $c$ for $f$ rounds before becoming \textit{boosting}. 
BFS($f, s$). Agents can *boosting*, *1/0-frozen* or *1/0-sensitive*.

- **Boosting**: Update their opinion with majority of their bit and the 2 bits they pull. If they see only agents of color $c$ for $s$ rounds, they become *$c$-sensitive*.

- **$c$-sensitive**: Turn into *$c$-frozen* if see value $c$.

- **$c$-frozen** keep value $c$ for $f$ rounds before becoming *boosting*. 
Thank You!