Fast, silent self-stabilizing distance-\(k\) independent dominating set construction\(^\star\)

Colette Johnen

Univ. Bordeaux, LaBRI UMR 5800, F-3340 Talence, France

Abstract

We propose a fast, silent self-stabilizing protocol building a distance-\(k\) independent dominating set, named \(\mathcal{FID}\). The convergence of the protocol \(\mathcal{FID}\) is established for any computation under the unfair distributed scheduler. The protocol \(\mathcal{FID}\) reaches a terminal (also legitimate) configuration in at most \(4n+k\) rounds, where \(n\) is the network size; it requires \((k+1)\log(n+1)\) bits per node.

keywords: distributed computing, fault tolerance, self-stabilization, distance-\(k\) independent dominating set, distance-\(k\) dominating set, distance-\(k\) independent set

1. Introduction

In this paper, we consider the problem of computing a distance-\(k\) independent dominating set in a self-stabilizing manner in the case where \(k > 1\). A set of nodes is a distance-\(k\) independent dominating set if and only if this set is a distance-\(k\) independent set and a distance-\(k\) dominating set. A set \(I\) of nodes is distance-\(k\) independent if every node in \(I\) is at distance at least \(k + 1\) to any other node of \(I\). A set of nodes \(D\) is distance-\(k\) dominating if every node not belonging to \(D\) is at distance at most \(k\) of a node in \(D\). We propose a very simple and fast protocol, called \(\mathcal{FID}\). The protocol \(\mathcal{FID}\) reaches a terminal

\(^\star\)This work was partially supported by the ANR project Displexity.

Email address: johnen@labri.fr (Colette Johnen)
URL: www.labri.fr/~johnen (Colette Johnen)
configuration in at most $4n + k$ rounds, where $n$ is the network size. \textit{FID} requires $(k + 1)\log(n + 1)$ bits per node. The obtained distance-$k$ independent dominating set contains at most $\lfloor 2n/(k + 2) \rfloor$ nodes.

**Related Works.** Silent self-stabilizing protocols building distance-$k$ dominating set are proposed in \cite{1,2}. These protocols do not build a $k$-independent set.

In \cite{3,4}, Larsson and Tsigas propose self-stabilizing $(l,k)$-clustering protocols under various assumptions. These protocols ensure, if possible, that each node has $l$ cluster-heads at distance at most $k$ from itself.

In \cite{5}, a silent self-stabilizing protocol extracting a minimal distance-$k$ dominating set from any distance-$k$ dominating set is proposed. A minimal distance-$k$ dominating set has no proper subset being a distance-$k$ dominating set. The protocol converges in $O(n)$ rounds, it requires at least $O(k \log(n))$ bits per node.

The paper \cite{6} presents a silent self-stabilizing protocol building a small distance-$k$ dominating set: the obtained dominating set contains at most $\lceil n/(k + 1) \rceil$. The protocol of \cite{6} converges in $O(n)$ rounds, it requires $O(\log(n) + k \log(n/k))$ bits per node. The protocol of \cite{7} builds competitive $k$-dominating sets: the obtained dominating set contains at most $1 + \lfloor (n - 1)/(k + 1) \rfloor$ nodes. The protocol of \cite{7} converges in $O(n)$ rounds; and it requires $O(\log(2k(2\Delta + 1).2n.D))$ bits per node, where $D$ is the network diameter, and $\Delta$ is a bound on node degree.

The protocols of \cite{7,6} use the hierarchical collateral composition of several silent self-stabilizing protocols including a leader election protocol and a spanning tree construction rooted to the elected leader. So their convergence time are larger than $4n + k$ rounds.

The presented protocol is simple: no use of the hierarchical collateral composition, no need for a leader election process, neither for a spanning tree building. Therefore, the protocol \textit{FID} is fast.
2. Model and Concepts

A distributed system $S$ is an undirected graph $G = (V, E)$ where the vertex set, $V$, is the set of nodes and the edge set, $E$, is the set of communication links. A link $(u, v) \in E$ if and only if $u$ and $v$ can directly communicate (links are bidirectional); so, the node $u$ and $v$ are neighbors. $N_v$ denotes the set of $v$’s neighbors: $N_v = \{ u \in V \mid (u, v) \in E \}$. The distance between the nodes $u$ and $v$ is denoted by $\text{dist}(u, v)$. The set of nodes at distance at most $k$ to a node $v$ is denoted by $k$-neighborhood$(v) = \{ u \in V \mid \text{dist}(u, v) \in [1, k] \}$.

**Definition 1 (distance-$k$ independent dominating set).** Let $D$ be a subset of $V$; $D$ is a **distance-$k$ dominating set** if and only if $\forall v \in V/D$ we have $k$-neighborhood$(v) \cap D \neq \emptyset$. Let $I$ be a subset of $V$; $I$ is a **distance-$k$ independent set** if and only if $\forall u \in I$ we have $k$-neighborhood$(u) \cap I = \emptyset$. A subset of $V$ is a distance-$k$ independent dominating set if this subset is a distance-$k$ dominating set and a distance-$k$ independent set.

To every node $v$ in the network is assigned an identifier, denoted by $id_v$. Two distinct nodes have distinct identifiers. It is possible to order the identifier values. The symbol $\perp$ denotes a value smaller than any identifier value in the network.

Each node maintains a set of shared variables. A node can read its own variables and those of its neighbors, but it can modify only its variables. The **state** of a node is defined by the values of its local variables. The cartesian product of states of all nodes determines the **configuration** of the system. The **program** of each node is a set of **rules**. Each rule has the form: $\text{Rule}_i : \langle \text{Guard}_i \rangle \rightarrow \langle \text{Action}_i \rangle$. The **guard** of a $v$’s rule is a boolean expression involving the state of the node $v$, and those of its neighbors. The **action** of a $v$’s rule updates $v$’s state. A rule can be executed by a node $v$ only if it is **enabled**, i.e., its guard is satisfied by the node $v$. A node is said to be enabled if at least one of its rules is enabled. A configuration is **terminal** if and only if no node can execute a rule.
During a computation step from a configuration one or more enabled nodes simultaneously perform an action to reach another configuration. A computation $e$ is a sequence of configurations $e = c_0, c_1, ..., c_i, ...$, where $c_{i+1}$ is reached from $c_i$ by a single computation step, $\forall i \geq 0$. A computation $e$ is maximum if it is infinite, or if it reaches a terminal configuration.

**Definition 2 (Silent Self-Stabilization).** Let $\mathcal{L}$ be a predicate on the configuration. A distributed system $S$ is a silent self-stabilizing system to $\mathcal{L}$ if and only if (1) all terminal configurations satisfy $\mathcal{L}$; (2) all computations reach a terminal configuration.

**Stabilization time.** We use the round notion to measure the time complexity. The first round of a computation $e = c_1, ..., c_j, ...$ is the minimal prefix $e_1 = c_1, ..., c_j$, such that every enabled node in $c_1$ either executes a rule or is neutralized during a computation step of $e_1$. A node $v$ is neutralized during a computation step if $v$ is disabled in the reached configuration.

Let $e'$ be the suffix of $e$ such that $e = e_1 e'$. The second round of $e$ is the first round of $e'$, and so on. The stabilization time is the maximal number of rounds needed by any computation from any configuration to reach a terminal configuration.

### 3. The protocol $FID$


**Notation 1.** A node $v$ is a head if $\text{dom}[0](v) = id_v$; otherwise it is an ordinary node.

Once the network is stabilized, any ordinary node $v$ has in its $k$-neighborhood a head having an identifier larger than its own identifier. And, the set of heads is a distance-$k$ independent set.
Protocol 1: FID: Fast distance-k independent dominating set construction

Shared variables

- \( \text{dom[]}_i(v) \) is a table of \( k + 1 \) members. A member is an identifier or \( \perp \).

Predicates

- \( \text{resignation}(v) \equiv \text{id}_v < \max \{ \text{dom}[i](v) \mid 0 < i \leq k \} \)
- \( \text{toUpdate}(v) \equiv \exists i \in [1, k] \text{ such that} \)
  \[ \text{dom}[i](v) \neq \max \{ \text{dom}[i-1](u) \mid u \in N_v \} \]
- \( \text{ordinaryToUpdate}(v) \) : \( \text{dom}[0](v) \neq \perp \)
- \( \text{headToUpdate}(v) \) : \( \text{dom}[0](v) \neq \text{id}_v \)

Rules

\( \text{RU}(v) : \text{toUpdate}(v) \rightarrow \)

for \( i \in [1, k] \) do \( \text{dom}[i](v) := \max \{ \text{dom}[i-1](u) \mid u \in N_v \} \); \n  if \( \text{resignation}(v) \) then \( \text{dom}[0](v) := \perp \); else \( \text{dom}[0](v) := \text{id}_v \);

\( \text{RE}(v) : \neg\text{toUpdate}(v) \land \neg\text{resignation}(v) \land \text{headToUpdate}(v) \rightarrow \)

\( \text{dom}[0](v) := \text{id}_v \);

\( \text{RR}(v) : \neg\text{toUpdate}(v) \land \text{resignation}(v) \land \text{ordinaryToUpdate}(v) ; \rightarrow \)

\( \text{dom}[0](v) := \perp ; \)

The value of \( \text{dom}[i](v) \) is \( \perp \) if there is no path of length \( i \) from a head to \( v \). Otherwise, the value of \( \text{dom}[i](v) \) is the largest head identifier such that there is a path of length \( i \) from this head to \( v \).

When an ordinary node \( v \) has no head in its \( k \)-neighborhood then the table \( \text{dom}[] \) in \( v \) does not contain any identifier. Notice that in this case, the predicates \( \neg\text{resignation}(v) \) and \( \text{headToUpdate}(v) \) are verified. So, the node \( v \) can perform the rule \( \text{RE} \) or the rule \( \text{RU} \). Hence, the set of heads is a distance-\( k \) dominating set in a terminal configuration.
The predicate \textit{resignation}(v) is verified when the node \(v\) has in its \(k\)-neighborhood a head \(u\) having an identifier larger than \(v\)'s identifier (i.e. \(id_v < id_u\)). If the node \(v\) is a head then the predicate \textit{ordinaryToUpdate}(v) is also verified. In this case, \(v\) can perform the rule \textit{RR} or the rule \textit{RU}. Therefore, the set of heads is a distance-\(k\) independent set in any terminal configuration.

The figure presents a terminal configuration of \(FID\) with \(k = 4\). The color of a node is the color of the head in its \(k\)-neighborhood having the largest identifier; the head identifiers are underlined. On each node, for \(0 \leq i \leq k\), the value of \(\text{dom}[i]\) is indicated unless it is \(\perp\). So, the table \(\text{dom[]}\) of node 82 contains the values (\(\perp, \perp, 70, 80, 90\)). Therefore, the node 78 has \(\text{dom}[3] \geq 70\) and \(\text{dom}[4] \geq 80\). As \(\text{dom}[4] \geq 80\), in the node 78; this node cannot become a head. The piece of information \(\text{dom}[3] \geq 70\), in the node 78 allows to the node 67 to be aware of the existence of the single head in its 4-neighborhood having an identifier larger than its own identifier.
4. Correctness of the protocol \(\text{FID}\)

In this section, we prove that the set of heads is a distance-\(k\) independent dominating set, in every terminal configuration of the \(\text{FID}\) protocol.

**Observation 1.** Let \(v\) be a node. In a terminal configuration, \(\text{dom}[0](v) = id_v \lor \text{dom}[0](v) = \perp\)

**Definition 3.** (\(\text{OrdinaryPr}(i)\)). For all \(i \in [1, k]\), the property \(\text{OrdinaryPr}(i)\) is defined as follow: if there is no path of length \(i\) from a head to the node \(v\) then \(\text{dom}[i](v) = \perp\); otherwise, \(\text{dom}[i](v) = id_u\) where \(id_u\) is the largest head identifier having a path to \(v\) of length \(i\).

**Lemma 1.** In a terminal configuration, the property \(\text{OrdinaryPr}(1)\) is verified.

**Proof.** According to observation [1] \(\text{dom}[0](u) \neq \perp\) if and only if \(u\) is a head \((\text{dom}[0](u) = id_u)\).

Let \(v\) be an ordinary node, in a terminal configuration. If \(v\) has no head in its neighborhood then \(\text{dom}[0](u) = \perp, \forall u \in N_v\). So \(\text{dom}[1](v) = \perp\). \(\perp\) is smaller than any identifier value. So, if \(v\) has a head in its neighborhood then \(\text{dom}[1](v) = \max \{id_u \mid u \in N_v \text{ and } \text{dom}[0](u) = id_u\}\).

**Lemma 2.** Let \(i\) be a positive integer strictly smaller than \(k\). In a terminal configuration, if the property \(\text{OrdinaryPr}(i)\) is verified then the property \(\text{OrdinaryPr}(i+1)\) is verified.

**Proof.** Let \(v\) be an ordinary node, in a terminal configuration in which the property \(\text{OrdinaryPr}(i)\) is verified. There is no path of length \(i+1\) from a head to \(v\) if and only if no neighbor of \(v\) has a path of length \(i\) to a head. We have \(\text{dom}[i](u) = \perp, \forall u \in N_v\). So \(\text{dom}[i+1](v) = \perp\).

Let \(w\) be the head having the largest identifier such that there is a path of length \(i+1\) from \(w\) to \(v\). \(v\) has a neighbor, denoted by \(u\), on its path to \(w\). As \(\text{OrdinaryPr}(i)\) is verified, \(\text{dom}[i](u) = id_w\), and \(\text{dom}[i](u') \leq id_w\) for any node \(u' \in N_v\). So \(\text{dom}[i+1](v) = id_w\).
**Theorem 1.** Let $c$ be a terminal configuration. In $c$, any ordinary node $u$ has a head in its $k$-neighborhood.

**Proof.** We will prove that if an ordinary node has no head in its $k$-neighborhood then the configuration $c$ is not terminal.

In $c$, for all $i \in [1, k]$, the property $\text{OrdinaryPr}(i)$ is verified by the lemma 1 and to the lemma 2. Let $u$ be an ordinary node without any head in its $k$-neighborhood. So there is no path of length lesser than $k + 1$ between $u$ and a head. We have $\text{dom}[i](u) = \bot, \forall i \in [0, k]$. So the predicate $\neg \text{resignation}(u) \land \text{headToUpdate}(u)$ is verified in $c$. The node $u$ can perform the rule $\text{RE}$ or the rule $\text{RU}$. ■

The following theorem establishes that the set of heads is a distance-$k$ independent set.

**Theorem 2.** Let $c$ be a terminal configuration. In $c$, a head has no head in its $k$-neighborhood.

**Proof.** We will prove that if a head has a head in its $k$-neighborhood then the configuration $c$ is not terminal.

Let $\text{wrongHeadSet}$ be the set of heads having one or more heads are in their $k$-neighborhood. Assume that $\text{wrongHeadSet}$ is not empty. Let $v1$ be the node of $\text{wrongHeadSet}$ having the smallest identifier. Let $v2$ be the closest head to $v1$. Let $d$ be the distance between $v1$ and $v2$. We have $0 < d \leq k$. According to the property $\text{OrdinaryPr}(d)$, $\text{dom}[d](v1) \geq id_{v2}$. So, in the configuration $c$, the predicate $\text{resignation}(v1) \land \text{ordinaryToUpdate}(v1)$ is satisfied. The node $v1$ can perform the rule $\text{RR}$ or the rule $\text{RU}$. ■

5. Termination of the protocol $\mathcal{FID}$

In this section, we prove that all maximal computations of $\mathcal{FID}$ protocol under the unfair distributed scheduler are finite by reductio ad absurdam arguments.

**Lemma 3.** Let $e$ be a computation of $\mathcal{FID}$ protocol under any scheduler. The computation $e$ has a suffix, named, $e'$ where no node changes its $\text{dom}[0]$’s value.
Proof. Assume that one or more nodes infinitely often modify their value of \( \text{dom}[0] \). Let \( \text{Set}^+ \) be the set of nodes that infinitely often modify their value of \( \text{dom}[0] \). We denote by \( u^+ \) the node of \( \text{Set}^+ \) having the largest identifier.

Let \( e_1 \) be the suffix of \( e \) in which no node having a larger identifier than \( u^+ \)'s identifier modifies its value of \( \text{dom}[0] \).

According to the definition of predicate \textit{resignation}, there is an integer \( i \) such that \( \text{dom}[i](u^+) > id_{u^+} \) infinitely often (at these times, \( u^+ \) becomes ordinary) and \( \text{dom}[i](u^+) \leq id_{u^+} \) infinitely often (at these times, \( u^+ \) becomes head). So \( u^+ \) has a neighbor named \( u_{i-1} \) such that (i) the value of \( \text{dom}[i-1](u_{i-1}) \) is infinitely often greater than \( id_{u^+} \) and (ii) the value of \( \text{dom}[i-1](u_{i-1}) \) is infinitely often smaller than \( id_{u^+} \). It is possible only if there is a path of \( i \) nodes, \( u_{i-1}, u_{i-2}, u_{i-3}, \ldots, u_0 \), such that (i) the value of \( \text{dom}[i-j](u_{i-j}) \) is infinitely often greater than \( id_{u^+} \) and (ii) the value of \( \text{dom}[i-j](u_{i-j}) \) is infinitely often smaller than \( id_{u^+} \) with \( 1 \leq j \leq i \). So, the value \( \text{dom}[0](u_0) \) is infinitely often greater than \( id_{u^+} \); and infinitely often smaller than \( id_{u^+} \). \( \text{dom}[0](u_0) \) can only take two values: \( \bot \) or \( id_{u_0} \). As \( \bot \) is smaller than any identifier value: \( u_0 \) has a larger identifier than \( u^+ \), and \( u_0 \) infinitely often changes its value of \( \text{dom}[0] \) during the computation \( e_1 \).

There is a contradiction. So, \( e_1 \) has a suffix \( e' \) where no node changes its value of \( \text{dom}[0] \).

**Lemma 4.** Let \( e \) be a computation of \( \mathcal{FID} \) protocol under any scheduler. The computation \( e \) has a suffix where no node changes any \( \text{dom}[i] \)’s values for \( 0 \leq i \leq k \).

Proof. The computation \( e \) has a suffix \( e' \) where no node changes its value of \( \text{dom}[0] \) (Lemma [3]).

For \( 0 < i \leq k \), let us name \( u_i \) a node that infinitely often modifies its value of \( \text{dom}[i] \) during the computation \( e' \). It is possible only if there is a path of \( i \)
nodes, \( u_{i-1}, u_{i-2}, u_{i-3}, \ldots, u_0 \), such that the value of \( \text{dom}[i-j](u_{i-j}) \) infinitely often changes, for \( 1 \leq j \leq i \). So, the value of \( \text{dom}[0](u_0) \) changes infinitely often during the computation \( e' \).

There is a contradiction.

In Lemma 4 we have established that any computation \( e \) has a suffix where all tables \( \text{dom}[\cdot] \) have their final values. Any action by any node \( v \) modifies a value of its table \( \text{dom}[\cdot] \). So, a terminal configuration is reached.

**Corollary 1.** Under any scheduler, all computations are finite.

### 6. Convergence time

In this section, we establish that the convergence time is at most \( 4n + k \) rounds.

**Lemma 5.** Let \( M \) bet the integer value of \( \max(n\lceil (k + 1)/2 \rceil^{-1}, 1) \). The size of a distance-\( k \) independent set is at most \( M \).

**Proof.** Let \( I \) be a \( k \)-independent set such that \( |I| > 1 \). Let \( v \) be a node of \( I \). We denote by \( \text{closest}(v) \) the set of nodes closer to \( v \) than any other node of \( I \). Notice that \( \bigcup_{w \in I} \text{closest}(w) \subset V \) and \( \text{closest}(v) \cap \text{closest}(u) = \emptyset, \forall (u, v) \in I^2 \). Let \( u \) be the closest node to \( v \) that belongs to \( I \). Let \( x \) be node on the path from \( v \) to \( u \) such that \( 0 \leq \text{dist}(v, x) \leq \lfloor k/2 \rfloor \). Let \( w \) be a node of \( I \) other than \( v \). We have \( \text{dist}(w, x) > k - \text{dist}(v, x) \geq \lfloor k/2 \rfloor \) because \( k < \text{dist}(w, v) \leq \text{dist}(v, x) + \text{dist}(x, w) \). So, \( \text{closest}(v) \) contains the first \( \lfloor k/2 \rfloor + 1 \) nodes in the path from \( v \) to \( u \). We conclude that \( 1 \leq |I| \leq n\lceil (k + 1)/2 \rceil^{-1} \). \( \blacksquare \)

**Notation 2.** Set \( 0 = \emptyset; V_i = V - S_{i-1}; \text{vh}_i \) is the node of \( V_i \) having the largest identifier; \( S_{i+1} = S_i \cup k\text{-neighborhood}(\text{vh}_i) \cup \{\text{vh}_i\}; T_i = 2i(k+1) \).

For all nodes \( u \), after the first round, the value of \( \text{dom}[0](u) \) is the identifier of a node in \( V \); this will stay true during the computation. For all nodes \( u \), after the second round, the value of \( \text{dom}[1](u) \) is also the identifier of a node in \( V \); this will stay true during the computation.
So, for all nodes \( u \), after the \( k + 1 \) first rounds, the table \( \text{dom}\[\] (u) \) contains only identifiers of nodes in \( V \); this will stay true during the computation.

After one more round, \( vh_0 \), the node having the largest identifier, is a head. It will remain a head during the computation (because \( \text{resignation}(vh_0) \) is never verified). After \( k \) more rounds, all nodes of \( k\text{-neighborhood}(vh_0) \), are and will remain ordinary because on these nodes, the predicate \( \text{resignation} \) remains verified forever.

So after \( T_1 = 2(k + 1) \) rounds, the nodes of \( \text{Set}_1 \) have their final status (ordinary or head).

After \( T_i + k + 1 \) rounds, for all \( l \in [0, k] \), we have \( \text{dom}[1](u_i) \in V_i \) for any node \( u_i \) of \( V_i \). This will stay true during the computation. So, after one more round, \( vh_i \) is a head; and it will remain a head.

After \( k \) more rounds, all nodes of \( k\text{-neighborhood}(vh_i) \), are and will stay ordinary (because, on these nodes, the predicate \( \text{resignation} \) remains verified forever).

So after \( T_{i+1} = 2(k + 1) + T_i \) rounds, the nodes of \( \text{Set}_{i+1} \) have their final status (ordinary or head).

The set \( HX = \{ v \mid \exists i \text{ such that } v = vh_i \} \) is a distance-\( k \) independent set. So \( V_M = \emptyset \).

We conclude that after at most \( 2n < T_M < 4n \) rounds, all nodes have their final status (ordinary or head). After \( k \) more rounds, in any node, the table \( \text{dom}[\] \) has its final values.

7. Reference


